Technical Report 1132

Improving Soldier Factors in Prediction Models

Rick Archer
Brett Walters
Alia Oster
Angela Van Voast
Micro Analysis and Design, Inc.

October 2002

United States Army Research Institute
for the Behavioral and Social Sciences

REPORT DOCUMENTATION PAGE

1. REPORT DATE (dd-mm-yy)	2. REPORT TYPE	3. DATES COVERED (from … to)
25 September 2002	Final	14 March 2000 – 30 April 2002

4. TITLE AND SUBTITLE	5a. CONTRACTOR GRANT NUMBER
Improving Soldier Factors in Prediction Models	DASW01-00-C-3011
	5b. PROGRAM ELEMENT NUMBER 665502

6. AUTHOR(S)	5c. PROJECT NUMBER
Rick Archer, Brett Walters, Alia Oster, & Angela Van Voast (Micro Analysis and Design, Inc.)	2O665502M770
	5d. TASK NUMBER M770
	5e. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
Micro Analysis and Design, Inc. 4949 Pearl East Circle Suite 300 Boulder, CO 80301	

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. MONITOR ACRONYM
U.S. Army Research Institute for the Behavioral and Social Sciences ATTN: TAPC-ARI-PO 5001 Eisenhower Avenue Alexandria, VA 22333-5600	ARI
	11. MONITOR REPORT NUMBER Technical Report 1132

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT *(Maximum 200 words):* Report developed under SBIR contract for topic A98-163. A key decision made at the highest levels of any military is the trade-off between allocating resources to system acquisition versus allocating resources to maintain force readiness through training. Advanced Distributed Simulations (ADS) provide a mechanism for tactical combat training through man-in-the-loop simulators and Computer Generated Forces (CGF). The potential for using ADS to address the trade-offs for allocating resources is dampened by the unrealistic behavior of CGF. Phase I of this project produced algorithms, data structures, and a methodology for incorporating the effects of training and environmental stressors to improve CGF behavioral realism. In Phase II, we expanded and enhanced the technical feasibility for including these effects in CGF entities on simulated battlefields. The resulting product is called the Training Effects and STressor Integration Module (TESTIM). It can provide the Army with the ability to improve the realism of CGF entities in ADS and other human performance models. TESTIM can also be used to assess the expected payoff of training in terms of improved performance.

15. SUBJECT TERMS
SBIR Report, Training, Computer-Generated Forces, Human Performance, Simulation, Stressors

SECURITY CLASSIFICATION OF			19. LIMITATION OF ABSTRACTION	20. NUMBER OF PAGES	21. RESPONSIBLE PERSON (Name and Telephone Number)
16. REPORT	17. ABSTRACT	18. THIS PAGE			
Unclassified	Unclassified	Unclassified	Unclassified	67	Dr. Stephen Goldberg (407) 384-3980

Technical Report 1132

Improving Soldier Factors in Prediction Models

Rick Archer
Brett Walters
Alia Oster
Angela Van Voast
Micro Analysis and Design, Inc.

U.S. Army Research Institute for the Behavioral and Social Sciences
5001 Eisenhower Avenue, Alexandria, VA 22333-5600

October 2002

Army Project Number **Small Business Innovation Research**
2O665502M770

FOREWORD

Semi-automated forces (SAF) are computer-generated entities used to represent enemy and friendly forces supporting the live players in a simulation. SAF originated with the Defense Advanced Project Agency's (DARPA) Simulation Networking (SIMNET) program. They dramatically reduce the number of players needed to participate in virtual and constructive training exercises. They provide soldiers an intelligent adversary or friendly subordinate or adjacent unit. Developers rely on computer science Artificial Intelligence methods to generate SAF behaviors. They result from applying extensive sets of if/then rules known as combat instruction sets. SAF behaviors for the most part are rule based and predictable. They tend to follow doctrine closely with few surprises.

An enemy, subordinate or adjacent friendly computer-generated force that demonstrates more variable behavior should benefit training. If the training audience can't easily predict what will happen, they will learn to prepare for unanticipated actions and react to them. They have to be prepared for unanticipated actions and react to them. The present research builds on earlier work begun by Dr. Philip Gillis to develop behavioral models, based on data and behavioral science theory, that could influence the behavior and performance of computer-generated entities in a combat simulation.

This report summarizes Phase I and Phase II of a Small Business Innovation Research project conducted by Micro Analysis and Design, Corp. of Boulder, Colorado.

The project was briefed to an audience consisting of representatives of the U.S. Army Simulation, Training, and Instrumentation Command's (STRICOM) Project Manager Warfighters' Simulation (PM, WARSIM), Project Manager Training Devices (PM,TRADE) and Engineering Directorate on April 11, 2002. There was a demonstration of how models representing training, aptitude, and sleep deprivation, running in an off-line server, could impact the behaviors and performance of OneSAF Testbed tank platoons.

STEPHEN L. GOLDBERG
Acting Technical Director

IMPROVING SOLDIER FACTORS IN PREDICTION MODELS

EXECUTIVE SUMMARY

Research Requirement:

In recent years, all branches of the military have been relying more heavily on simulation exercises for training soldiers, planning military operations, and as analysis tools for allocating budget resources. These simulation exercises often have human and non-human participants [generally referred to as Computer Generated Forces (CGF)]. A problem with CGF is that they often do not behave as realistically as human participants would. They are programmed to use doctrinally correct behavior reflecting soldiers that are fully trained under "normal" battlefield conditions. CGF entities do not behave differently when they have been working for long periods of time without sleep or under extreme environmental conditions. There is also currently no way to adjust CGF behaviors to reflect the different levels and types of training the soldiers may have received or their different aptitudes. As a result, simulation exercises are not as effective as they could be.

There are several factors that affect human performance; levels of training, environmental stressors, and soldier aptitude are a few. Research into the impact these factors have on human performance is increasing our understanding of these effects at a rapid rate. If the constructive simulations that the military is using for soldier training, operations planning, and resource projections are to be most effective, they need to be able to take advantage of the latest findings from the human performance research community. However, to rewrite the constructive simulations every time a new human performance effect is understood and quantified would be so inefficient that new research would rarely be incorporated into the simulations. What is needed is a tool that effectively and efficiently integrates realistic human performance variability into the behaviors of CGF entities in these simulations.

Procedure:

This project had three major objectives. The first was to determine the effects of training and environmental stressors on human performance variables. The second objective was to find a quantitative way to calculate these effects. The third objective was to find an efficient way to incorporate these performance effects into the CGF behaviors of constructive simulations.

To meet these objectives, several different algorithms needed to be either developed or found in the literature. These included a training effects algorithm, algorithms for calculating the effects of aptitude and experience on performance, environmental stressor algorithms, and a methodology for combining the effects of each performance variable. The search for these algorithms began with a review of the literature. At the same time, a survey was developed and used to collect data directly from soldiers in the field. The survey asked the soldiers to estimate the effect that their own training had on their performance.

The simulation platform selected as the target environment for testing and implementing these algorithms was the OneSAF Testbed Baseline (OTB) Version 1.0. Rather than coding these algorithms directly into the OTB software, an alternate approach was implemented. This approach used a client-server architecture and placed the algorithms in a server external to the client CGF.

Findings:

During our literature review, we found an extensive amount of research on the effects of training. Four distinct learning curves were identified that are commonly accepted in the training and skill acquisition community: the power law, the exponential, the hyperbolic, and the logistic curves. After the survey data were collected from soldiers, a mathematical methodology was developed to combine the data into learning rates and then fit to match both the power law curve and the exponential curve. Other algorithms were also developed to model the effects of aptitude, five environmental stressors (cold, fatigue, heat, mission oriented protective posture, and noise), and experience on performance. The software module that was developed to calculate all of these performance effects is called the Training Effects and STressor Integration Module (TESTIM).

The OTB software was slightly modified to add a graphical user interface (GUI) that allows an OTB user to specify, for a selected entity, the amount and type of training that the entity has received for each of the taxonomic categories identified through the research. The user also specifies the crewmembers' initial proficiency prior to the training and which stressors will be in effect during a simulation run.

For each simulation, the input values from OTB are passed to a Soldier Performance Server (containing TESTIM) via a Middleware software module that keeps track of which OTB entity is requesting the information. The Soldier Performance Server contains a number of human performance models - each representing an OTB behavior. The Middleware determines which model should execute for each performance request. As the model executes, it communicates with TESTIM for needed stressor effects calculations (the training effects were all calculated prior to the run). When the Soldier Performance Server model completes its execution, the requested performance value is sent back to OTB.

Utilization of Findings:

This server approach has significant advantages over embedding code for human behavior in OTB. First, it is not invasive to OTB. The Soldier Performance Server models can be modified to reflect new human performance research without requiring OTB changes. Models can be of varying types ranging from simple discrete event simulations to complex cognitive models.

Second, this approach creates a number of opportunities for synergy between the Human Systems Integration (HSI) and training communities, and virtual simulation developers. The HSI community has been developing human performance models of military operations for several years, many of which have been validated. If an extensive list of behaviors from

simulations like OTB were developed, the wealth of HSI models could be used as a starting point for Performance Server models rather than starting from scratch.

In addition, most HSI models have been designed and developed to help analysts identify "high driver" behaviors, that is, behaviors that are critical determinants of total system performance. Training personnel and commanders who develop OTB scenarios to train soldiers can use this information to insure that the scenarios include practice on these critical tasks. Another opportunity for synergy exploits man-in-the-loop simulation as a data collection opportunity. By collecting data on exactly what the human operator does during the simulation, network models of task performance could be extracted from the data for building additional Soldier Performance Server models. This information could be passed on to the HSI community to help improve their models and build new ones, ultimately supporting the use of quantitative analyses to support system acquisition.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

Introduction

A key decision made at the highest levels of any military is the choice between allocating resources to system acquisition versus allocating resources to maintain force readiness through training. Furthermore, these issues arise not only at the highest levels of resource allocation, but also system design. How do we trade off design expectations with training demands? These resource allocation questions are increasingly being answered with computer modeling and simulation. However, there are virtually no models that predict the effects of training or the combined effects of environmental stressors on soldier performance. Until this deficiency is rectified, we can expect that decisions will be made that do not correctly value an investment in training. What are needed, therefore, are models that link training and environmental stressors to performance in combat and combat support operations.

In December of 1998, Micro Analysis and Design, Inc. (MA&D) was awarded a Phase I SBIR entitled "Improving Soldier Factors in Prediction Models." The overall goal of the Phase I effort was to investigate algorithms, data structures, and a methodology for incorporating the combined effects of training and environmental stressors into a tool that could be used to influence human performance models. The main purpose for developing this tool was to improve the realism of computer generated forces (CGF) in distributed simulations. For Phase II of this project, we enhanced these algorithms and data structures and programmed them into the Training Effects and STressor Integration Module (TESTIM).

Phase I Summary

This section briefly summarizes the work that was performed under Phase I. For a complete description of all the work performed, see Archer, Walters, Yow, Carolan, and Laughery (1999). Six tasks were performed to accomplish the overall goal. First, we developed a model of the basic relationships between training and performance to be modeled in CGF. Next, we reviewed the relevant literature and ongoing projects in order to develop better algorithms to address the interactions amongst environmental stressors. Third, we developed data structures and analytical methods within an available human performance modeling tool to support the relationships between training, environmental stressors, and performance. These data structures were then populated with expert opinion data. Empirical data sources were also identified to replace some of the estimated data during Phase II. Finally, we identified a target environment for embedding these models into CGF in Phase II. The remainder of this section describes the results of these tasks.

Modeling Training Effects

One of the main focuses for Phase I of this SBIR was to develop a model that included algorithms and data structures for including the effects of training in human performance models. Our strategy was to spend Phase I defining this structural model of training effects based upon a review of the literature and interviews with experts. Only by developing a structural model and the list of variables to be included could a focused search for applicable data be made.

The challenge of this effort was to generalize training effectiveness across a variety of tasks. Unlike environmental stressors such as heat or fatigue, training variables can vary for different kinds of tasks. For example, if it is hot or if a soldier has gone a long time without sleep, the amount of heat stress and fatigue are the same for all of the tasks that the soldier performs, even though the environmental stressors affect different tasks in different ways. With training, we need to consider *how much* training and *what kind* of training the soldier has received for each task. A goal of this project was to generalize the effects of several training variables across all of the types of tasks that soldiers perform.

The first step in generalizing training effects across a variety of soldier tasks was to develop a taxonomy or set of categories that could characterize combat tasks. Work that has been done for environmental stressors such as heat, cold, fatigue, chemical agents, etc. have developed similar taxonomies for soldier tasks that include categories such as visual perception, auditory perception, fine and gross motor skills, and acts of cognition. These are skill categories that may be degraded as a result of one or more environmental stressors.

We quickly found ourselves struggling in our attempts to describe the effects of training on a skill like visual perception or acts of cognition. We then decided to develop a taxonomy that was representative of the kind of skills that soldiers are trained on for combat missions. Our initial taxonomy contained eight skill categories for training. Our approach was to review the usefulness and completeness of this initial taxonomy and make necessary revisions to it, as we elicited the subject matter expert (SME) estimates and reviewed the empirical data necessary to implement the training algorithms and data structures. The taxonomy was refined and now includes the following:

1. Command and control
2. Communications
3. Planning
4. Tactics and doctrine
5. Technical proficiency with equipment

The next step in the development of the training algorithms and data structures was to identify the training variables that we wanted to include. The number of variables that exist in the training literature is far too extensive to include practically in our algorithm. The goal was to select some meaningful variables that could 1) have some theoretical basis, 2) be complex enough to include important aspects of training, 3) be simple enough to be understandable and doable, 4) be generalizable to all taxon categories, and 5) affect both time and accuracy of human performance.

The first variable we included in the algorithm is *type* of training. The training types we included were classroom, simulator, and field training. We also included a variable for the amount of each training type and the maximum amount that each training type could contribute to peak performance. Training types and amounts could be mixed to achieve either peak performance or some percentage of peak performance. For example, a user could define that the CGF could have some amount of classroom training, some other amount of simulator training and an additional amount of field training, none of which by itself would result in peak

performance. Our data structure included a maximum percentage of peak performance that could be achieved by any one training type for any one training taxon category. By this we mean that one type of training such as classroom may only be able to bring a trainee up to some percentage of peak performance. In this case, supplementary training of another type would be necessary to reach peak performance. In addition to a determination of the maximum contribution to peak performance for any one training type, we included the concept of a minimum requirement. By this we mean that there may be some tasks that must have a minimum amount of one or more training types before peak performance can be reached. Finally, we used the following equation (Hancock & Bayha, 1992; Newell & Rosenbloom, 1981) to calculate task times and accuracies (i.e., the amount of performance attained) given the actual amount of training:

$$Y = KX^{-A}$$

Where:

Y = task time (or accuracy)
K = worst case performance time (or accuracy)
X = amount of training
A = training constant.

The simulation software *Micro Saint* was chosen as the human performance modeling tool to test our initial methodology. The scenario chosen to demonstrate the capabilities of the developed algorithms was a simple movement to contact mission that included platoon-maneuvering, engagement with several enemy patrols, and the occupation and defense of positions. At that time, estimated values for all of the variables in our algorithms were used.

After we developed and tested this methodology and began our search for empirical data on the effects of training in the Option phase, we quickly realized that it would be nearly impossible to find data in the format we needed. Likewise, we felt that it would be difficult for trainers or soldiers to provide estimates for these values. Our methodology for generating these learning curves was therefore changed in Phase II. Because of this, no further discussion on the Phase I approach will be given. Also, note that the results of our literature review on the effects of aptitude, training, forgetting, experience and environmental stressors on performance are discussed in a separate section.

Potential Data Sources

Another focus for Phase I was to identify empirical data sources that might replace some or all of the estimated data during Phase II. We initially proposed to populate the data structures of our training algorithm with expert opinion data. These data were intended to serve as preliminary input to our algorithms. The SME data could then be replaced as empirical data became available, depending on the estimated validity of the SME data and the cost associated with collecting empirical data. Five potential empirical data sources were identified and reviewed that we believed might eventually replace the SME data:

1. Army training plan and curriculum documents
2. Published research studies

3. Performance data from the Combat Training Centers
4. Home station training records
5. The long-term evaluation of the Close Combat Tactical Trainer (CCTT)

Target Environment

The final task for Phase I was to identify a target environment for embedding the training and stressor models in order to influence the behavior of CGF. For the Phase I demonstration, we selected a *Micro Saint* network model of a tank platoon movement to contact scenario. For Phase II we identified two target environments. The first was the OneSAF Testbed Baseline Semi-Automated Forces (OTB SAF) CGF Version 1.0 package developed for the U.S. Army. It allows the user to create and control entities on a simulated battlefield. These entities replicate the SAFsim and SAFstation. These components are typically run on separate computers distributed over a network, although the SAFsim and SAFstation can run on the same computer. The components communicate physical battlefield state and events among themselves through the Distributed Interactive Simulation (DIS) protocol and command, control, and system information through the Persistent Object (PO) protocol.

The other platform that we identified as a target environment was the Command, Control, and Communications Simulation (C3SIM) evaluation test-bed. Due to time constraints, only OTB was actually used in Phase II.

Phase II Objectives

The primary goal of the Phase II effort was to expand upon the technical feasibility for including training effects and environmental stressors in CGF entities on synthetic battlefields. To accomplish this goal, we identified four technical objectives:

1. Improve and expand the training effects and combined stressor algorithms demonstrated in Phase I

2. Integrate TESTIM into targeted CGF environments and run experiments to assess the functionality and validity of the performance effects produced by TESTIM

3. Estimate the validity of input data, training algorithms, and demonstration results

4. Analyze targeted training data sources to replace some or all of the estimated data obtained from the SMEs during the Phase I Option task

Based on the four technical objectives listed above, we conducted the following six tasks.

Task 1. First, we added an aptitude dimension to the training and stressor algorithms. Aptitude can have an impact on performance through a number of different mechanisms. For example, there is evidence that general aptitude translates into more proficient learning skills and therefore faster learning (Gottfredson, 1997; Sticht, Armstrong, Hickey, & Caylor, 1987). General ability has also been shown to predict job performance, and in more complex jobs it does so better than any other single personal trait, including education and experience (Mchenry,

Hough, Toquam, Hanson, & Ashworth, 1990). These findings, along with others, are discussed in the literature review section of this report.

Task 2. Next, as part of our on-going effort to make human performance and CGF models more realistic, we added performance decay functions (based on time since training) to our training algorithm. We felt that a model that included these functions would be a more powerful tool for considering how training can benefit performance.

Task 3. For our third task, we developed a graphical user interface to support the input of data required by TESTIM. This interface, described in more detail later in this report, allows a user to choose from default values for the required training data structures. The user can also modify these values to reflect different experimental conditions. Furthermore, the interface can be used to indicate which tasks will be modified in the CGF models and to specify the taxonomic categories that each of those tasks will fall into.

Task 4. We integrated TESTIM with a simple OTB scenario of a tank platoon engagement via a Soldier Performance Server. This task was aimed at incorporating improved decision making capabilities and modifying the physical performance characteristics of OTB entities. Two examples of physical human performance tasks are loading a weapon system and engaging an enemy target. TESTIM modifies the physical human performance tasks by affecting the time it takes the OTB entities to perform these tasks. The decision-making capability of OTB entities was also changed such that less predictable and more human like decision-making representations were made. Another aspect of this demonstration was the use of OTB environmental variables as input to the environmental stressors that we included in the TESTIM. OTB continually updates a number of variables that reflect the current status of the battlefield environment. These variables include things like wind speed and direction, cloud cover, rainfall, temperature, and relative humidity. The server approach is described in detail later in this report.

Task 5. For the fifth task of Phase II, we measured the face validity of TESTIM's algorithms using results from the scenario developed in Task 4. That is, we tested whether the results from TESTIM and the OTB scenario matched what we expected to observe. Whether the algorithms make use of SME data or empirical data, a measure of the validity will determine how much confidence we can place in results obtained using these algorithms. We ran simulation trials varying the amounts of training on each skill and training type and collected resulting performance measures.

Task 6. For the final task, we analyzed existing empirical data sources to replace some or all of the estimated data obtained from the Phase I Option task. After a review of the sources identified in Phase I, as well as numerous research reports, we were only able to obtain empirical data on the effects of gunnery training on performance (e.g., Graham & Smith, 1990; Hart, Hagman, & Bowne, 1990; Hoffman & Melching, 1984; Kraemer & Smith, 1990; Morrison, Drucker, & Campshure, 1991; Turnage & Bliss, 1989). Most of the training research involved very simplistic tasks or non-Army tasks that did not relate to our taxonomy. In those few cases where specific Army tasks were studied, the dependent variables that were monitored could not

be related to the information we needed for our algorithms. Therefore, we did not replace the estimated data.

As part of our statement of work, we also proposed to integrate the algorithms and data structures developed in Phase I with algorithms that have been developed by the Army Research Institute as a part of the development of the Human Performance Model (HPM) and C3SIM (for a review, see Gillis & Hursh, 1999). As mentioned in the Phase I summary, C3SIM and the HPM were not used due to time constraints. Furthermore, the algorithms used in the HPM to model the effects of experience and aptitude did not fit well with our data structures. For these reasons, we did not incorporate them into TESTIM.

Literature Review

A major component of both Phase I and II of this project was an extensive analysis of the literature from several different research areas. The literature from industrial and educational psychology, education, and training were all mined for relevant reports. Additionally, the Defense Technical Information Center reports were searched for any Department of Defense funded research that may link performance improvement with training. It was hoped that a thorough review of relevant literature would enable us to construct algorithms that represent the specific amount of performance improvement or decrement that can be linked to specified amounts of training, time since training, stressors, aptitude, and experience.

Skill Acquisition

Our literature review revealed several theories on skill acquisition that have been developed to explain learning on different types of tasks. These theories attempt to explain the relationship between training and performance by curve fitting. In a review of nearly 3,000 research titles and abstracts, Lowry, Rappold, and Copenhaver (1992) identified four major learning curves: the power law, hyperbolic, exponential, and logistic curves.

The power law is one of the most extensively supported learning curves. It fits a wide variety of skill data such as perceptual-motor skills, motor skills, and cognitive skills. Prior to Newell and Rosenbloom's extensive research on the power law in 1981, skill acquisition research was dominated by exponential, hyperbolic and logistic functions. All of these shapes are still found in current learning curve literature. Of these three shapes, the most prevalent and the main contender against the power law is the exponential function. When data are fitted with both power and exponential curves, it is often difficult to distinguish which is the "better" fit by conducting a simple examination of the curves (Lane, 1986). Generally an extensive statistical analysis is required to determine which function more accurately models the data.

The hyperbolic curve is a special case of the power law and it tends to fit empirical data almost as well as the power law does (Lane, 1986). On the other hand, the logistic curve has a unique shape. It produces S-shaped or sigmoid curves. This shape is usually found when the empirical data are performance ratings provided by instructors (Lane). Since the hyperbolic curve is a special case of the power law and the logistic curve does not fit most empirical data we chose not to use them as models for our survey data.

We chose to concentrate our efforts on fitting our SME data with the power law curve and the exponential curve. There are two key differences between the power law curve and the exponential curve. First, the exponential curve increases more rapidly than the power law curve; consequently it has a steeper slope (see Figure 1). Second, in the power law the time spent training on a task (T) is directly related to the resulting proficiency on the task (see Figure 2). With the exponential law, the relationship between time spent training and resulting proficiency is indirect.

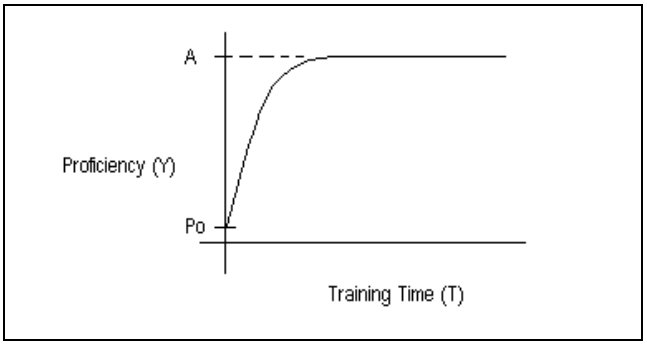

Figure 1. Exponential learning curve.

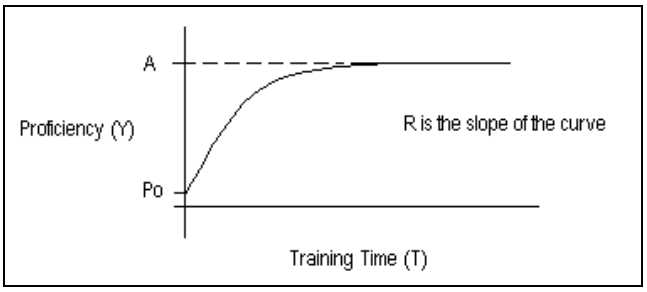

Figure 2. Power law curve.

Both the power law curve and the exponential curve are increasing functions with negative acceleration. Negative acceleration means that as the time spent training increases the relative benefit of the training decreases. We will be using the following equations and notations for the two curves:

$Y = A - (A - P_o)(T + 1)^{-R}$ (Power function)
$Y = A - (A - P_o)\, e^{-RT}$ (Exponential)
Where:

T = time spent training
Y = proficiency level after training
R = rate of learning
A = highest level of proficiency that can be attained with training (asymptote)
P_o = proficiency on 1st trial (i.e. innate proficiency/ worst case).

Bayesian approach. As part of our review, we also considered the use of probabilistic reasoning using Bayesian networks as a potential modeling framework. Bayesian network technology is a recently developed technique for speeding up the process of doing probabilistic reasoning with large complex relational networks. Using Bayesian networks for diagnostic assessment and profiling trainee proficiency is a rapidly developing area of research (e.g., Nichols, Chipman, & Brennan, 1995). Bayesian networks have been primarily used in learning research as a mechanism to diagnose the knowledge, skills, and/or strategies that are used in solving a particular problem, making a decision, or performing an action. However, they could in principal be used to infer or predict expected performance based on patterns of training and practice.

There are a number of reasons why we decided against using Bayesian networks to model the relationship between training and performance. These reasons include the following: 1) the power law has a strong empirical research base and theoretical framework, 2) Bayesian modeling requires defining prior probabilities encoding the hypothesized relationship between training and performance (this would involve more time and resources for SME time and for model validation), and 3) the power law fits well with the task oriented approach of task network modeling tools such as *Micro Saint* and with the Army's Mission Essential Task List (METL) oriented approach to planning training.

Retention

Another important factor in determining the effects of training on performance is the amount of time that has passed since the training was received. The rate at which skills are forgotten is often characterized as a negatively accelerating function based on time since training. But unlike the literature on skill acquisition, no general curves on retention (decay) are widely accepted. Decay data reported in the literature vary widely in terms of the time span over which they occur; measured in terms of seconds, minutes, hours, days, weeks, months, and years. Some researchers agree that decay can be represented as a power function similar to the ones used to estimate the rate of learning, and they have attempted to quantify this function (e.g., Rose, Czarnowlewski, Gragg, Austin, Ford, Doyle, & Hagman, 1985). However, results from long-term research have lead Bahrick and Hall (1991) to conclude that although a significant amount of what influences decay is determined by acquisition processes, acquisition performance alone is not an accurate predictor of decay.

Research has shown that the rate at which skills and knowledge decay can depend on factors such as the amount and quality of training received at the time of initial learning. Skills that are trained to automaticity (mastery training) tend to decay at a slower rate than tasks trained to just proficiency. Also, the methods used to assess decay can suggest different decay rates. For instance, Luh (1922) used five different measures to index decay: recognition, reconstruction, written reproduction, recall by anticipation, and relearning. Luh found that while decay functions for all measures were negatively accelerated, they had different rates of decline. Finally, while some skills fall sharply during the time immediately following acquisition and decline more slowly as additional time passes, other skills do not begin to show decay until several months after acquisition. For example, perceptual-motor skills (e.g., driving, flight control, and sports skills), show very little forgetting over long periods of time. In contrast,

procedural skills, which require a sequence of steps, such as how to use a text processor or how to run through a checklist for turning on a piece of equipment, tend to be quickly forgotten (Rose, 1989). For these reasons, we chose to concentrate our efforts on fitting our SME data with a negatively accelerating function similar to the power function (just as was done for the learning curves).

Environmental Stressors

Other variables that are used in our algorithms to make soldiers' behaviors in CGF models more realistic are environmental stressors. Performance on a task or mission under ideal conditions may differ drastically from performance on the same task or mission under stressful conditions. The environment in which military operations (tasks and missions) are conducted can be very stressful. Incorporating stressors into simulation models allows users to estimate mission performance under "worst case" conditions.

A review of the relevant literature produced performance degradation factors for the following stressors: heat, cold, noise, fatigue, circadian rhythm, Mission Oriented Protective Posture (MOPP gear), and altitude (Bradley & Robertson, 1998; Micro Analysis and Design Incorporated & Dynamics Research Corporation, 1999; Walters, French, and Barnes, 2000). However, some of these degradation factors have been correlated to affect a specific set of taxonomical categories (e.g., visual, numerical, cognitive skills and other taxons). Still, these taxons are fairly generalizable and users should be able to relate their own taxons to these taxons.

Empirical and theoretical research were also examined on the combined effects of multiple stressors and how to model them. Although an abundant amount of empirical data has been collected on the separate effects of stressors on performance, there has only been a small amount of work done on the *combined* effects of *some* stressors. Therefore, difficulties occur when equations are developed that try to generalize the interactive effects of multiple stressors. Two different human performance modeling tools were identified that contain equations that address the combined effects of stressors: the Integrated Performance Modeling Environment (IPME) (Micro Analysis and Design Incorporated, 1999) and the Improved Performance Research Integration Tool (IMPRINT). Both of these tools have equations that are reasonable for modeling many interactions amongst stressors. However, they may not be reasonable for all interactions. Note that our work is not dependent on the IPME or IMPRINT tools for functionality. Rather, they are sources for equations that attempt to explain the complex relationship between multiple stressors and performance. For now, we have decided to use the following equation (Harris, 1985) to combine the effects of multiple stressors:

$$DF_T = \Pi\, i = 1, n \sqrt[i]{DF_i}$$

Where:

DF_T = Total degradation factor

DF_i = The i^{th} ordered degradation factor

n = Number of degradation factors.

Using the above equation, when two or more stressors are combined, the overall degradation is *less* than the sum of the individual degradations. The most severe stressor will have a full effect on performance. As additional stressors are added, they will have less impact on performance.

Aptitude

Our review of the literature also revealed that aptitude has a significant effect on performance, especially as it relates to the level of performance that can be achieved with a specified level of training. For example, a 1969 study done for the U.S. Army found that individuals in the bottom 20% of their ability distribution required up to five times as much instruction and practice to attain minimal proficiency in basic military tasks such as rifle assembly (for a review, see Gottfredson, 1997; Sticht, Armstrong, Hickey, & Caylor, 1987). General ability has also been shown to predict job performance, and in more complex jobs it does so better than any other single personal trait, including education and experience (Gottfredson, 1997). For example, the Army's Project A, a study conducted in the 1980s to improve the recruitment and training process, found that general mental abilities were highly correlated with technical proficiency and soldiering proficiency (Mchenry, Hough, Toquam, Hanson, & Ashworth, 1990).

For this project, the effects of aptitude were calculated for task times and decision accuracy using data from Project A. An important step in our approach is to assign an entity's aptitude level. A general aptitude factor can be extracted from skill batteries such as the Armed Forces Vocational Aptitude Battery (ASVAB), intelligence category (CAT) and standard intelligence tests (Gottfredson, 1997). The ASVAB is a battery of tests administered to enlistees prior to entering the military services. In this project, an entity's aptitude level can be assigned in one of two ways: 1) selecting the Military Occupational Specialty (MOS) of the entity (soldier) performing the task or 2) selecting the minimum ASVAB cutoff score of the soldier that can perform the task (this is described further in the OTB Interfaces section of the report).

In our approach, each task is weighted based on how much of the following taxons (skills) are required to perform the task: visual, numerical, cognitive, fine motor discrete, fine motor continuous, gross motor light, gross motor heavy, communications (read/write), and communications (oral) (This is because the effects of aptitude vary depending on the task). Then, the aptitude of the entity being modeled is compared to the aptitude for the MOS 12B (baseline), which is the designation for a tank Platoon Leader. Based on this difference and the task-to-skill weightings, a multiplication factor is pulled from a lookup table (generated from Project A data) and used to change the task time or decision accuracy. If the aptitude of the entity is the same as the baseline, then the task time (accuracy) does not change. If the aptitude of the entity is less than the baseline, then the task time increases (or the accuracy decreases). If the aptitude of the entity is greater than the baseline, then the task time decreases (or the accuracy increases).

Experience

Several researchers have investigated the relationship between experience and environmental stressors. Some of the findings are summarized by Hancock (1986). In 1962, Fitts argued that training should continue past an often arbitrary and minimal criterion of

performance since this increases resistance to stress, fatigue and interference. Mackworth (1950) found that less experienced workers suffered more disruption by increasing heat stress, and this was manifested earlier than for skilled workers. Similar results were found for several different types of tasks: Morse code message transmission, Naval lookout duty, and physical exercise. Blockley and Lyman (1951) found the same pattern of results for flight performance under heat stress. In a seven-year research project called TADMUS (Tactical Decision Making Under Stress), researchers (sponsored by the Office of Naval Research) studied and developed guidelines on training, simulation, decision support, and display principles to mitigate the impact of stress on decision-making. Cannon-Bowers and Salas (1998) outline the overall background, research approach, and paradigm used by TADMUS. One of their major findings and assertions is that exposing trainees to stress can inoculate them from the impact of stress, and that highly competent individuals and teams will be more resilient to the effects of stressors. Unfortunately, we were not been able to find useful empirical data for our algorithms from which to affect performance based on experience. We did, however, feel it was important to include experience within TESTIM. Therefore, a simple performance multiplier was put in TESTIM along with the capability for users to enter their own experience effects.

Empirical Data

Our literature review also focused on finding empirical data for specific Army tasks. Gunnery training was chosen because of the relatively standard way in which performance is measured. All empirical data reviewed on gunnery training dealt with the effects of *simulator* training on performance. Research has consistently shown positive effects for several different types of simulators [e.g., the Institutional Conduct of Fire Trainer (I-COFT), the Videodisk Gunnery Simulator (VIGS), and TopGun]. Gunnery skills improve during training on simulators and these skills have been shown to transfer to the real world (e.g., Abel, 1986; Sterling, 1996; Turnage & Bliss, 1989). Unfortunately, no data has been found on the effects of actual live-fire gunnery training or classroom training on gunnery performance. Furthermore, as Morrison, Drucker, and Campshure (1991) have noted, there has been little research done that has directly addressed the effects of training on the *retention* of gunnery skills.

Much of the research discussed here may hold promise for developing algorithms to impact performance of CGF. However, several obstacles exist to the creation of these algorithms based on the data found in the studies reviewed. First, the studies described here primarily use comparative statistics to evaluate statistically different scores on pre- and post-training measures of performance or effectiveness. While these statistics provide support for the worthiness and effectiveness of the training interventions, they say little about how much improvement can be expected following the training event. Second, few of the studies provide enough detail regarding the duration of the training event, or the training event is a long term effort which confounds the ability to isolate the effects of training from those of learning. Finally, most of the measures of performance used to evaluate training effectiveness are outcome measures. That is, most measures take the form of an overall achievement score on some dependent variable prior to and after exposure to training. While such measures may offer insight into the amount of overall improvement that can be attributed to training, they say little about the specific impacts and what aspects of performance are responsible for increases in overall scores. To be most useful to efforts for encoding the effects of training on performance, the tasks under investigation

should be granular enough to allow for specific explanation of performance changes (e.g., reduced time to perform the task, increased decision accuracy, etc.).

Data Collection

As an alternative to the unavailable empirical data relating the effects of training on soldier performance, we collected SME estimates of training effects on performance. In order to make the process of obtaining these estimates as easy and non-intrusive as possible, we developed a questionnaire asking soldiers about their own training and their perception about how that training affected their performance. This questionnaire is shown in Appendix A.

We believe estimates from SMEs are valid sources of data for several reasons. First, a study performed for the Nuclear Regulatory Commission (Engh, Yow, & Walters, 1998) investigated how well SME data on task times and workload values compared to task times and workload values from an experiment performed in a high fidelity nuclear plant simulator. Over 70% of the task times from the experiment fell within a 90% confidence interval of the data collected from SMEs. This substantial agreement between SME data and empirical data produces confidence that, if collected properly, SME data can be valid.

A second point of confidence in SME data comes from standard practices in the simulation industry. It is an accepted procedure to use SMEs to judge the validity of the output of simulations. Proceedings from the SIMVAL99 Conference held in January (Glasow & Pace, 1999) stated that SMEs are an essential part of simulation validation. We concluded that if SMEs are judged to be accurate enough to *validate* our simulations, then their data is accurate enough to be used as *input* to our simulations.

In order to eventually generalize the training effects data to a wide variety of soldier tasks, we wanted to obtain training and performance data on general categories of tasks that soldiers are trained on and are familiar with. As a result, we chose the tasks in the Mission Training Plan (MTP) for Armor platoons (ARTEP 17-237-10) as the focus for our SME data collection effort. The MTP is categorized into seven Battlefield Operating System (BOS) categories containing a total of 38 tasks. MTP tasks represent broad categories of soldier tasks that can be decomposed into increasing levels of detail that cover the majority of combat tasks that soldiers are trained to perform.

To obtain the best and most recent estimates, we requested troop support from field units. In November and December of 2000, we interviewed soldiers at Fort Riley, Kansas and Fort Carson, Colorado to obtain training effects estimates. At each location, we interviewed eight Platoon Leaders and eight Platoon Sergeants for a total of 32 interviews. From these interviews we obtained the data we needed to develop the learning curves described later in this report. During the interviews, we had the soldiers complete our questionnaire. In the questionnaire, the soldiers were asked about the training they received on the 38 tasks from the ARTEP MTP and on tank gunnery. The primary focus of the questionnaire was how much and what kind of training the soldiers had received over the last year and how this training had affected their performance.

In order to fit the SME data to both the power law and exponential curve, we needed to determine the highest level of proficiency that could be attained on a task with the training type in question. This highest level of proficiency is referred to as the asymptote. Usually the asymptote occurs naturally within the data. In this data collection effort, there was a high probability that the asymptote would not emerge from the data set because we were only looking at one year of a soldier's training, not his entire career. Another reason that an asymptote may not have emerged from the data is that Army tasks are typically trained to criterion and not to mastery level. To compensate for this, we asked the soldiers to estimate the highest level of proficiency they could reach with each type of training for each of the 38 tasks.

In addition to the asymptote for each type of training for each task, we also needed to determine a soldier's innate proficiency on a task. A soldier's innate proficiency is the level of proficiency with which the soldier could perform a task during his first attempt. It was assumed that the soldier had performed the task for the first time after some minimal or basic training. Because it cannot be guaranteed that the soldiers had performed each of the 38 tasks for the first time in the last year, we asked them to provide us with an estimate of their innate proficiencies.

There were two different parts to our questionnaire. The first part dealt with innate proficiency and asymptotic proficiencies for each task. The second part of the questionnaire required them to tell us how much training they received in the last year. To help us collect decay information, we broke the last year into quarters. For each quarter the soldiers provided us with number of hours of training and proficiencies. Each soldier provided us with the following data for each of the 39 tasks in the survey:

1. Innate proficiency
2. Asymptote for classroom training
3. Asymptote for simulator training
4. Asymptote for field training
5. Beginning proficiency for the first quarter
6. Ending proficiency for all quarters
7. Number hours of classroom training during each quarter
8. Number hours of simulator training during each quarter
9. Number hours of field training during each quarter

Of the 32 questionnaires collected, more than half of them had to be removed from the final data set. A large percentage of the questionnaires were removed because of the soldiers' misinterpretation of the questions. For example, the most common problem with the results was that the soldiers wrote the same number of hours down for every single task in a single quarter. Our assumption is that they did not have the time to think about how much training they had had on each specific task. For our purposes, this type of data was of no use because specific training data was needed to calculate the learning curve parameters. Another problem that we ran into with some of the platoon leaders' data was that they had not yet received a full year of training at the time the data was collected.

Algorithm Development

The first step in creating the taxon learning curves was to convert the data that we collected from the SMEs (which was at the task level) to the taxon level. To accomplish this conversion, the SMEs were asked to decompose each task into our five taxons. This information was given in percentages. For example, the MTP task *Assault an Enemy Position (17-3-0220)* could be broken down into the taxonomy as follows:

- Planning 12.5%
- Communication 15%
- Command and control 35%
- Technical proficiency with equipment 15%
- Tactics and doctrine 22.5%

Converting Survey Task Data to Taxon Data

Using the percentages described above, we initially attempted to perform the conversion from task to taxon data using what is called the least squares method (for a review, see Wackerly, Mendenhall III, & Scheaffer, 2002). Unfortunately, this method resulted in a loss of detail and produced a tremendous amount of statistical error in the SME data. In turn, numerous questionnaires no longer provided useful proficiency data for a taxon. Additionally, the taxon data that could be used resulted in several learning rates that appeared to be too slow. We concluded that the SME data at the task-level contained too many errors for the least squares method to provide useable data.

The alternate method chosen to convert the task data to taxon data was mathematically less rigorous than the least squares method. We applied SME task-taxon mappings to determine which tasks had the highest percentages of the different taxons in order to generate the taxon-level data.

Creating Learning Curves from Taxon Data

Using the taxon-level data, we created learning curves for each training type and taxon in our taxonomy. The first learning curve to which we fit our data was the power curve. The power law that we defined earlier has three parameters that were derived from the taxon-level soldier data:

1. Asymptote (A)
2. Innate proficiency (P_o)
3. Learning rate (R)

Asymptotic proficiencies (A) and innate proficiencies (P_o) were taken directly from the taxon data; the learning rates (R) were calculated. In our questionnaire, we asked the soldiers for the number of hours they spent training (for each training type) over the past year. We also asked for the beginning proficiency and the ending proficiency for each quarter of the past year.

All of these values, along with asymptotes and innate proficiencies, were used to calculate the learning rates for each taxon in our taxonomy.

Embedded Learning Curve

The major challenge in calculating the learning rates was to determine how much each training type contributed to the overall change in the soldiers' proficiencies from quarter to quarter. We only knew that the soldiers had *x* hours of classroom training, *y* hours of simulator training, and *z* hours of field training that resulted in a percentage increase in his or her proficiency for the taxon. We did not know how much of that percentage increase could be attributed to the classroom, simulator, or field training. Because we did not have this information, we could not solve for the different training type learning rates individually.

To deal with this problem, we developed a nonlinear equation (see Equation 1, $h(r)$) that combined the three different power curves (one for each training type). This equation allowed us to calculate the learning rates simultaneously. The final equation is shown below:

$$h_1(\vec{r}) = a_3 - (a_3 - p_o)\left(t_3 + \left(\frac{a_2 - (a_2 - p_o)\left(t_2 + \left(\frac{a_1 - (a_1 - p_o)\left(t_1 + \left(\frac{p_b - a_1}{p_o - a_1}\right)^{-1/r_1}\right)^{-r_1} - a_2}{p_o - a_2}\right)^{-1/r_2}\right)^{-r_2} - a_3}{p_o - a_3}\right)^{-1/r_3}\right)^{-r_3} \quad (1)$$

In this equation, the variables are:

- a_1, a_2, a_3 = Asymptote 1st training type experienced, asymptote 2nd training type, asymptote 3rd training type
- p_o = Innate proficiency
- p_b = Beginning proficiency
- t_1, t_2, t_3 = Time spent in 1st training type, time spent in 2nd training type, time spent in 3rd training type
- r_1, r_2, r_3 = Learning rate for 1st training type, learning rate for 2nd training type, learning rate for 3rd training type.

This equation is an embedded power curve. The outside of the curve is simply the power curve for the third type of training. However, instead of a factor of 1 being added to the T (t_3) value, there is another equation being added to the T value.

$$g(\vec{r}) = \left(\frac{a_2 - (a_2 - p_o)\left(t_2 + \left(\frac{a_1 - (a_1 - p_o)\left(t_1 + \left(\frac{p_b - a_1}{p_o - a_1} \right)^{-1/r_1} \right)^{-r_1} - a_2}{p_o - a_2} \right)^{-1/r_2} \right)^{-r_2} - a_3}{p_o - a_3} \right)^{-1/r_3} \quad (2)$$

Equation 2 ($g(r)$) represents the amount of time it would take a soldier to reach the proficiency that the soldier reached with the second type of training using the third type of training. Within this equation, there is another equation (see Equation 3, $j(r)$) with the power curve form and a different equation being added to the T value (t_2). The equation being added to the T value is the amount of time it would take for the soldier to reach the proficiency the soldier reached with the first type of training if the soldier had the second type of training.

$$j(\vec{r}) = a_2 - (a_2 - p_o)\left(t_2 + \left(\frac{a_1 - (a_1 - p_o)\left(t_1 + \left(\frac{p_b - a_1}{p_o - a_1} \right)^{-1/r_1} \right)^{-r_1} - a_2}{p_o - a_2} \right)^{-1/r_2} \right)^{-r_2} \quad (3)$$

There is one more equation with the power curve format in this combined learning curve; it is the power curve for the first type of training a soldier receives. In Equation 4 ($k(r)$), the T value (t_1) does not have a factor of 1 added to it. Instead, there is a ratio raised to some power added to the T value. This is the amount of time it would have taken the soldier to reach his beginning proficiency for the quarter (p_b) with only the first training type. In other words, we do not know how much time it really took the soldier to reach p_b, but we assumed that the soldier only had one type of training up to that point. With this assumption, we can determine how long it took the soldier to reach that proficiency based on his innate proficiency (p_o). If the soldier's beginning proficiency for the quarter is equal to the soldier's innate proficiency (i.e. it was the soldier's first time to perform the task), then the ratio simplifies to 1. Then this equation simply reverts to the general power curve.

$$k(\vec{r}) = a_1 - (a_1 - p_o)\left(t_1 + \left(\frac{p_b - a_1}{p_o - a_1} \right)^{-1/r_1} \right)^{-r_1} \quad (4)$$

Solving a System of Embedded Learning Curves

The embedded learning curve shown in Equation 1 accounts for multiple learning types, which means we can solve for all the learning rates, for a particular taxon, simultaneously. To solve an equation with multiple unknown variables, it is necessary to have a system of unique equations. If the system has the same number of equations as there are unknown variables, then standard numerical methods can be used to solve for the system's unknown variables. If there are more equations in the system than unknown variables, there is a high probability that the system is inconsistent. Below is a simple example of inconsistency for a system of several equations with only one unknown variable.

$2x = b_1$
$3x = b_2$
$4x = b_3$

This system is only solvable if the right-hand sides are in the ratio 2:3:4 (Strang, 1988). If the right-hand sides are not in this ratio, then the system is typically considered to be unsolvable. The best way to deal with inconsistent systems is "to choose the x that minimizes the average error in the m equations" (Strang).

A popular definition for the average error is the sum of the squares. The sum of squares is defined as follows for the linear system $ax = b$:

$$\phi = \sum_{i=1}^{m} \left\| (a_i x - b_i) \right\|^2$$

When there is an exact solution to the system, then the error ϕ is zero. In most cases, there will not be an exact solution and the error function will be a parabola. The minimum of a parabola is when the first derivative of the function equals zero. So the solution to the system $ax = b$ that produces the minimum average error is when

$$\frac{d\phi}{dx} = (a_1 \bar{x} - b_1)a_1 + \ldots + (a_m \bar{x} - b_m)a_m = 0$$

This equation simplifies to $\bar{x} = \frac{a^T b}{a^T a}$ and is referred to as the least squares solution (Strang, 1988). This idea can be applied to inconsistent linear systems of m equations with n unknown variables. The least squares solution for a system with multiple unknowns is very similar to the solution for a system with one unknown, $\bar{x} = (A^T A)^{-1} A^T b$.

In the case of solving for three learning rates for a particular taxon, we are dealing with an inconsistent system; for each taxon there are approximately one hundred embedded learning curve equations. Therefore, we needed to use a least squares method to manipulate each system before we could calculate the learning rates.

The primary difference between the embedded learning curve systems and the system used to explain the least squares method is that the embedded learning curve is nonlinear and the example is linear. Our system of nonlinear equations has the form

$f_1(r_1,r_2,\ldots,r_n) = b_1$
$f_2(r_1,r_2,\ldots,r_n) = b_2$
.
.
.
$f_m(r_1,r_2,\ldots,r_n) = b_m$

The error function for this system is $\phi(\vec{r}) = \sum_{i=1}^{m} (f_i(\vec{r}) - b_i)^2$, and like the linear system, when the error function $\phi(\vec{r}) = 0$, there is an exact solution to the system. If we defined a new nonlinear system $\vec{G}(\vec{r})$ where $\vec{G}_i(\vec{r}) = f_i(\vec{r}) - b_i$ then the error function can be simplified and written in vector notation $\phi(\vec{r}) = \vec{G}(\vec{r})^T \vec{G}(\vec{r})$.

For the linear system, the least squares solution is when the first derivative of the error function equals zero. In the nonlinear case, the least squares solution is when the gradient of the error function equals zero. The gradient of the error function simplifies as follows:

$$\nabla\phi(\vec{r}) = \nabla(\vec{G}(\vec{r})^T \vec{G}(\vec{r}))$$
$$\nabla\phi(\vec{r}) = 2 * J(\vec{r})^T \vec{F}(\vec{r})$$

Where $J(\vec{r})$ is the Jacobian matrix of $\vec{G}(\vec{r})$.

To find the least squares solution to an inconsistent nonlinear system, we must find when $\nabla\phi(\vec{r})$ equals zero. Specifically, when this is true for our embedded learning curve system, then r contains the learning rates for each training type for a taxon.

Numerically Solving the Embedded Learning Curve Systems

Once our embedded learning curves systems were manipulated using the least squares methodology, we needed to determine which numerical method would give us the most accurate learning rates. There are several different methods available for finding the zeros (minimums) of a nonlinear system of equations. The two primary numerical methods are Newton/quasi-Newton methods and gradient methods. Newton and quasi-Newton methods for nonlinear systems usually converge quickly; unfortunately, they are not guaranteed to converge if the initial starting point is poor. Gradient methods converge more slowly than the Newton and quasi-Newton methods, but they are usually guaranteed to converge to the correct answer even when a poor initial guess is provided (Burden & Faires, 1997).

After testing both Newton and gradient numerical methods, we decided to use a gradient method. There were two different gradient methods that we tested: the Steepest Descent method and the Conjugate Gradient method. Both of these methods utilize the least squares methodology for nonlinear systems. The difference between these two methods is that the Steepest Descent method uses first-order derivates while Conjugate Gradient uses second-order derivates (Aoki, 1971). In general, using second-order derivates accelerates the method's rate of convergence.

The Conjugate Gradient method requires calculating a matrix referred to as a Hessian matrix. For the algorithm to work correctly, the Hessian matrix must be positive-definite. If the matrix is not positive-definite, the method will not converge as expected (Aoki, 1971). For our purposes, there was no guarantee that the Hessian matrix of embedded learning curves would be positive-definite. Therefore, the Conjugate Gradient method would not work.

The Steepest Descent method does not converge as quickly as the Conjugate Gradient method, but it does not require the calculation of a Hessian matrix. After thoroughly testing this the Steepest Descent method, we found that it calculated the correct learning rates with three decimal places of accuracy. To obtain this level of accuracy requires a large number of iterations and a very low tolerance (i.e., tolerance = 1e-20). We, therefore, used the Steepest Descent method to calculate the best approximations to the learning rates for each taxon in our taxonomy. These calculated learning rates were considered to be the learning rates for the average soldier. To get the innate proficiency and asymptotes for the average soldier for each taxon, we calculated the averages using the taxon-level data. We ended up with an embedded learning curve for each taxon.

Performance Decay Functions

The data collected from the questionnaire also served as input for building functions that represent a decrease in proficiency for periods of time soldiers do not receive training for particular tasks. The methodology for developing these decay functions was similar to that of the learning curves.

Test-Bed Model

The purpose of the test-bed model was to provide us with a way to test our training and stressor algorithms (i.e., our calculator) before we tried to integrate them with OTB. This test-bed model allowed us to ensure that our calculator produced significant effects on performance. It was also helpful in testing the communication links between our calculator and *Micro Saint* (the simulation software used to develop the model).

Our intent was to have the calculator affect task times and task accuracies within the test-bed model. For example, the calculator could affect how long it takes to track a target in the "Attack" network of the test-bed model. It could also affect the probability of a fire hitting a target or the probability of spotting an opponent. In addition, we wanted to affect decisions that are made within the model (e.g., the decision of whether to use smoke to confuse the enemy).

Test-Bed Model Scenario

The scenario modeled was a U.S. tank platoon conducting a hasty defense. It was based on an example scenario described in the book *Modeling Human and Organizational Behavior: Application to Military Simulations* (Pew & Mavor, 1998). The example scenario is a detailed vignette of a hasty defense that involves a platoon of M1 tanks defending a battle position. The outcome of the vignette is that the U.S. tank platoon defeats the enemy and they successfully defend the battle position.

It was our intent to use the vignette as a basis for our model. The primary difference between the vignette and the actual test-bed model was that we wanted to add variability. We wanted there to be the possibility that the enemies did not always come from the same direction and that the US tank platoon leader and sergeant did not always make correct decisions. In the end, we wanted to be able to affect the performance of both the U.S. tanks and the enemy tanks based on the soldiers' training.

One of the components of the vignette that we inserted into the test-bed model was that the enemy would make its initial approach from one of three directions. Once the enemy made its initial approach, additional groups of enemy tanks would come to help in the attack (i.e., two separate groups of T-80 tanks and some BMPs[1]). Another component of the vignette was that the U.S. tanks would have artillery and attack helicopters to back them up in their hasty defense. The platoon leader could ask for these when he thought it was time to use them. Ideally, the platoon leader should use the artillery to destroy tanks that were in the initial approach. Then, when the platoon leader has determined that his forces have slowed down the enemy sufficiently, he can ask for permission to retreat back to the Main Company Battle Position.

Figure 3 illustrates the task network diagram of the test-bed model. An animated version of the activities occurring in the model was also developed and can be seen in Figure 4. Performance was measured in the test-bed model environment by the dependent variables (model output) listed below:

- Mission success – a successful mission is one in which the enemy retreats and the friendly forces remain in their position
- Decision to request helicopter support
- Decision to request artillery
- Decision to shoot smoke for camouflage

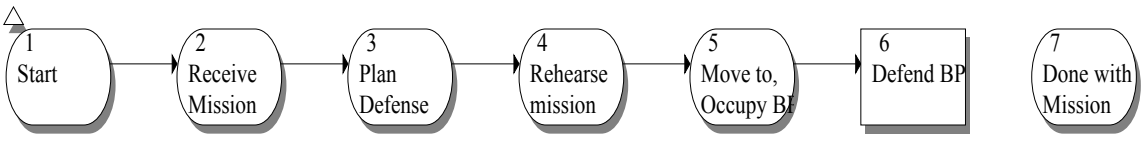

Figure 3. Top-level network of the test-bed model.

[1] BMP = boyevaga mashina pakhoty, a Soviet armored personnel carrier.

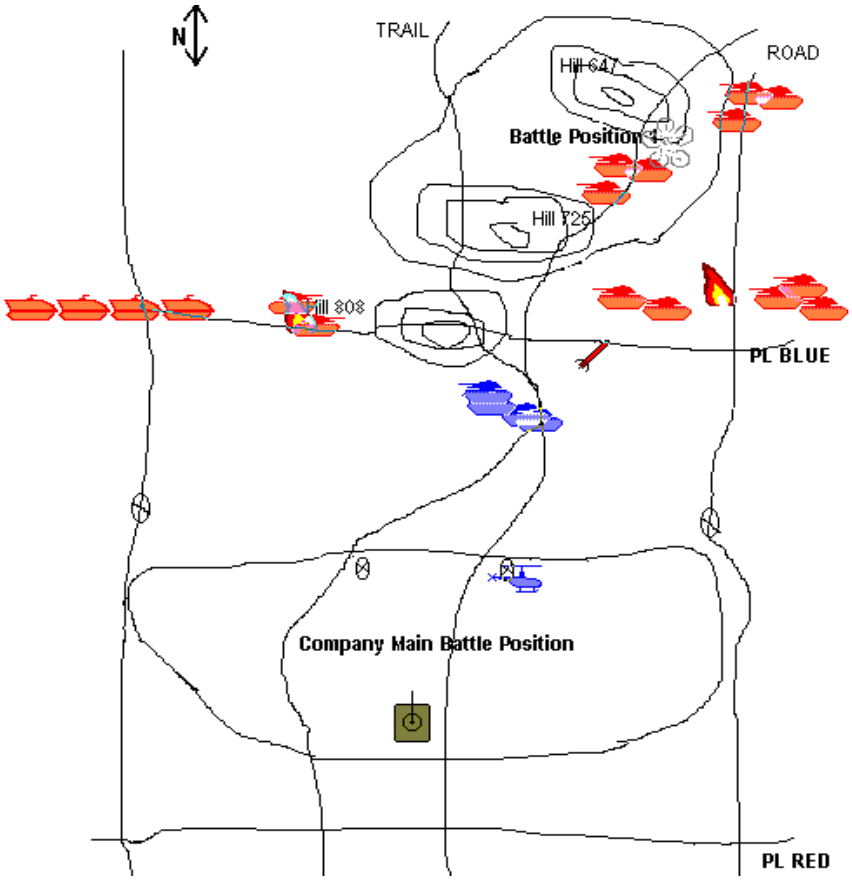

Figure 4. Animated view of the test-bed model during a simulation run.

TESTIM Software

Another focus for Phase II of this SBIR was the development of a software tool that allows users to select or develop training taxonomies, training types, and parameters for learning, decay, stressors, and experience. This tool, called TESTIM, resides on the same computer as the Performance Server in a Microsoft Windows operating system. Figure 5 shows the main interface of TESTIM. This is the first interface a user sees when the tool is started.

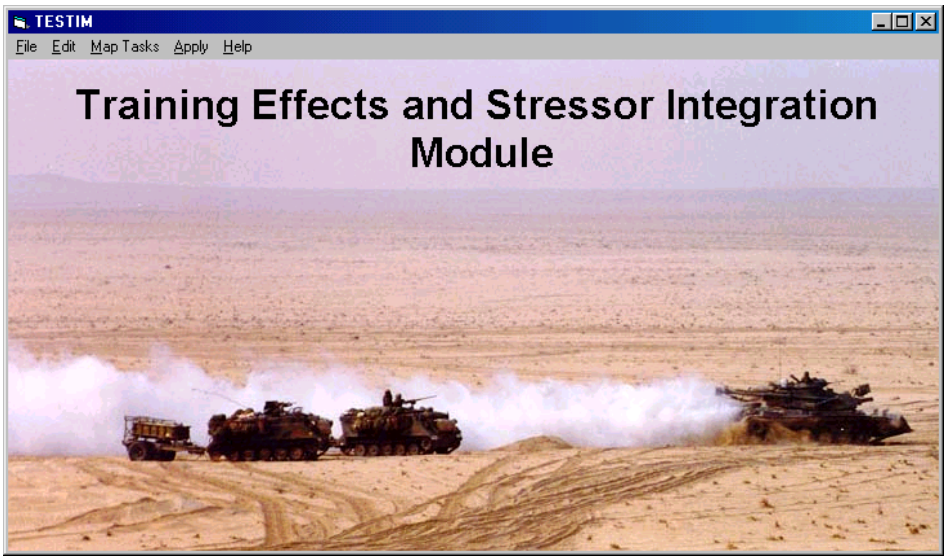

Figure 5. TESTIM main interface.

File Menu. A user begins using TESTIM by selecting either a library or a previously developed user performance effects set by selecting the "Open" option of the File menu. A performance effects set consists of a training taxonomy, training types, learning curve and decay parameters, stressors, experience, and aptitude modifiers. Once, selected, the "Open" option displays a list of the currently saved performance effects sets including the default library set. Figure 6 shows the "Open Performance Effects Set" interface. When a set is selected, a temporary working copy of the selected set is created. All of the changes that a user makes to any aspect of the selected set are actually made to the working copy.

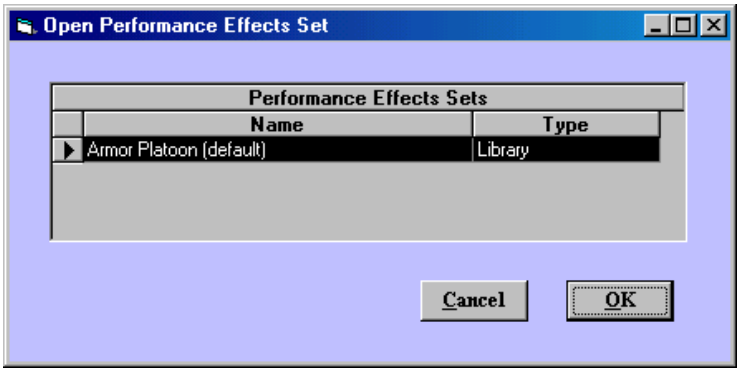

Figure 6. TESTIM interface to select a performance effects set.

When the working copy of a performance effects set has been modified, the changes can be made permanent using the "Save" command. However, changes cannot be made to the default Library set. When the default Library set is the selected set, the "Save" command is disabled.

The "Save As" command saves the changes made to the working copy of the currently selected set with a different file name. Users can open a library or user performance effects set, make changes to the working copy, and then save the changes as a new performance effects set using the "Save As" command.

The "Exit" command will close the TESTIM development software tool. If changes have been made to the currently selected performance effects set since the most recent save, a warning will be displayed.

Edit menu. The Edit menu allows users to modify one or more components of an existing performance effects set. In order to edit an existing set, the user must first open the set from the File menu. When there is no set currently open, the options on the Edit menu are disabled. To cancel changes that have been made to the current set, users can either open a different set or select the "Exit" command from the File menu.

The "Training Taxons" menu option allows users to change the name of existing taxons, add new taxons, or delete taxons. Selecting the "Training Taxons" option will display a list of the training taxons stored in the current performance effects set as shown in Figure 7. The "Cancel" button returns the user to the main TESTIM interface without committing any changes that were made to the current data set. The "OK" button saves the current items in the taxon list to the working copy of the current set. If any changes are made to the training taxons in the current performance effects set these changes must be saved (using the "Save" or "Save As" File Menu) before the "To Training Taxons" menu option of Map Tasks menu can be used.

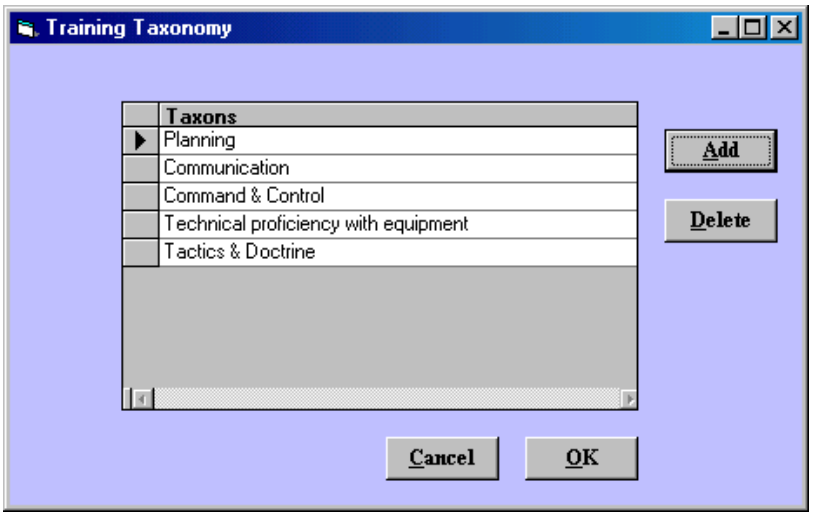

Figure 7. TESTIM interface to edit list of taxons.

The "Training Types" menu option allows users to change the name of existing training types, add new training types, or delete training types. Selecting the "Training Types" option will display a list of the training types stored in the working copy of the current performance effects set as shown in Figure 8.

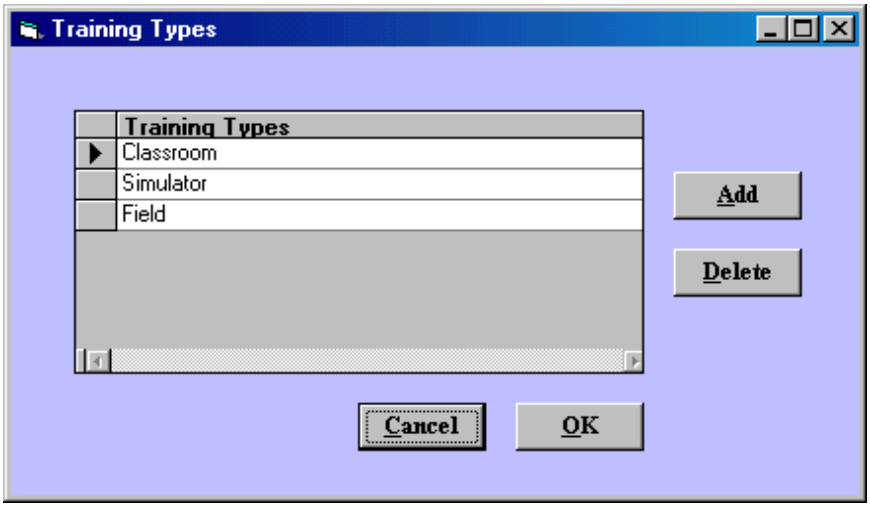

Figure 8. TESTIM interface to edit list of training types.

The "Learning Curve Parameters" option, shown in Figure 9, allows users to enter parameters required to calculate learning curves for each taxon by training type. The drop-down menu box at the top-left of the interface allows users to choose a training type from the list of training types for the current effects set. The table (grid) in the center of the interface shows the learning curve parameters for the current effects set. The leftmost column of the grid displays the taxon list in the current effects set. This is a locked column and cannot be modified from this interface. The two fields to the right of the taxon list allow users to edit the proficiency asymptote and learning rate for the learning curve function.

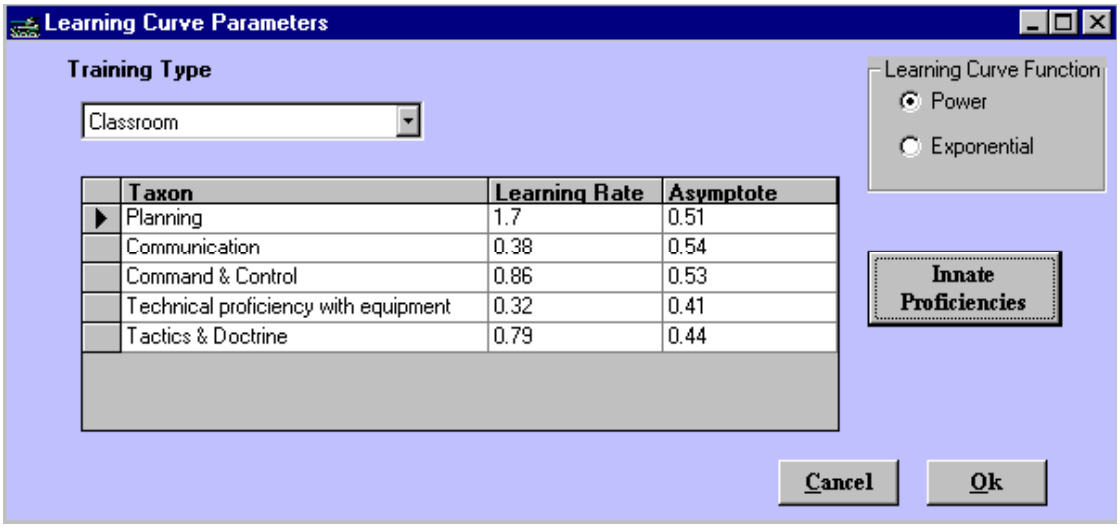

Figure 9. TESTIM interface to edit learning curve parameters.

The default library performance effects set uses a power function for the learning curve. However, users that have their own data will be able to select either a power function for the learning curve of an exponential function. If the power function is selected, the user will need to enter an innate proficiency for each training taxon using the interface shown in Figure 10. The left column showing the taxon list is locked and cannot be modified from this interface. Innate proficiency is defined as a soldier's proficiency without any formal training.

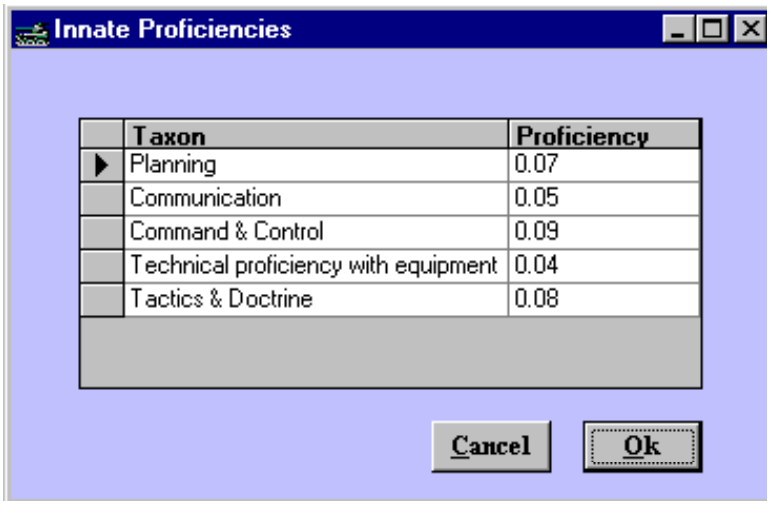

Figure 10. TESTIM interface to edit innate proficiencies.

The "Decay Parameters" option (see Figure 11) allows users to enter decay rates for each taxon. The column displaying the taxons is locked and cannot be modified from this interface.

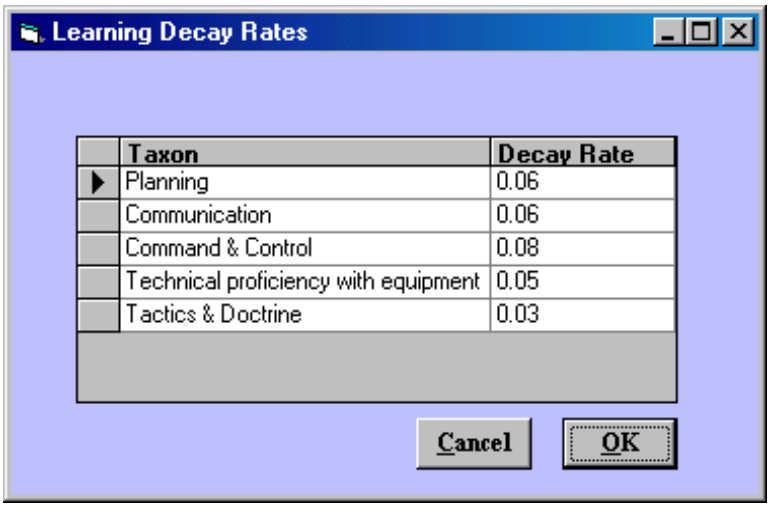

Figure 11. TESTIM interface to edit decay rates.

The "Stressors" option on the Edit menu contains four submenu options. Each option allows users to edit multipliers for task times and accuracies for each stressor taxon. Figure 12 shows the interface for editing the "heat" stressor multipliers. Users cannot edit the "Relative

Humidity" column or any of the column headers. Currently, there are some taxons for which we do not have data. These tables are populated with 1.00 to indicate no performance decrement.

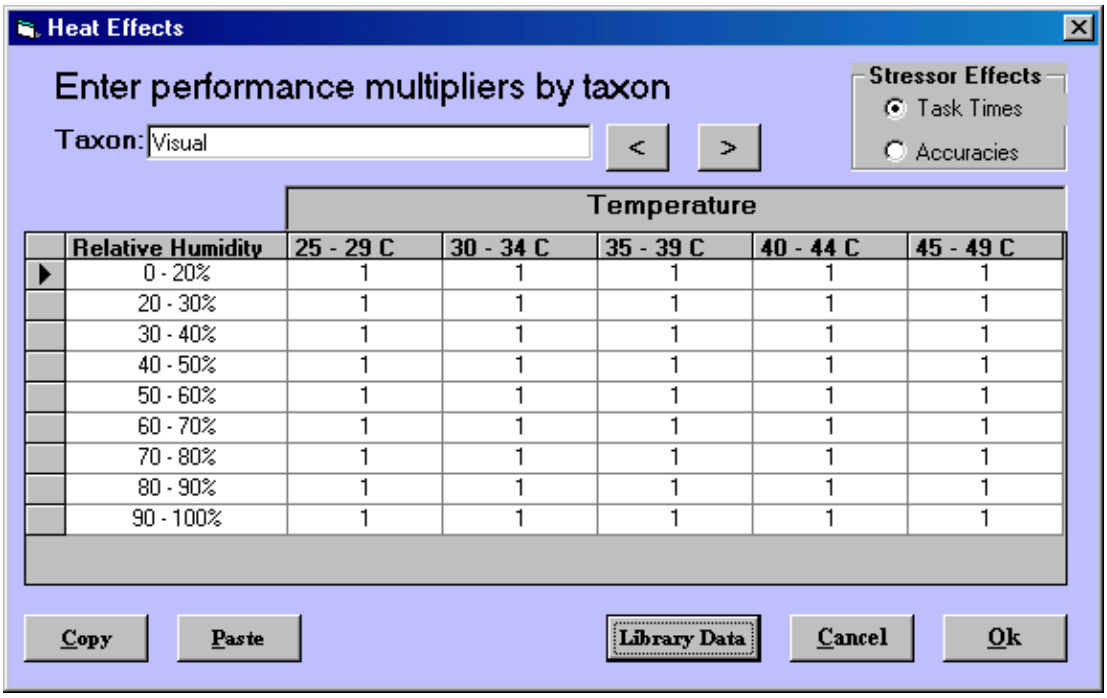

Figure 12. TESTIM interface to edit the heat stressor look-up table.

The "Stressor Effects" frame in the top-right corner of the interface allows users to switch between task time tables and the accuracy tables. If a user has modified some or all of the cells in a table and wishes to start over with the library data, he or she can do so by clicking the "Library Data" button. The "Cancel" button will return the user to the main TESTIM interface without making any changes to the working copy of the selected effects set. The "OK" button will save the changes to the working copy of the effects set and return the user to the main interface. In addition, a user can copy rows of data and paste the data into other rows using the "Copy" and "Paste" buttons. If a user wishes to copy multiple rows at a time he must use the "Ctrl" key to do this. If the user picks a row in the middle of his row set then the entire set will be unselected. To paste, the user must select the same numbers of rows as he copied using the "Copy" button.

Figure 13, Figure 14, and Figure 15 show look-up tables used to degrade task times due to the effects of cold temperatures, noise, and MOPP gear, respectively. As with the heat effects look-up table, users cannot edit the leftmost column data or any of the column headers.

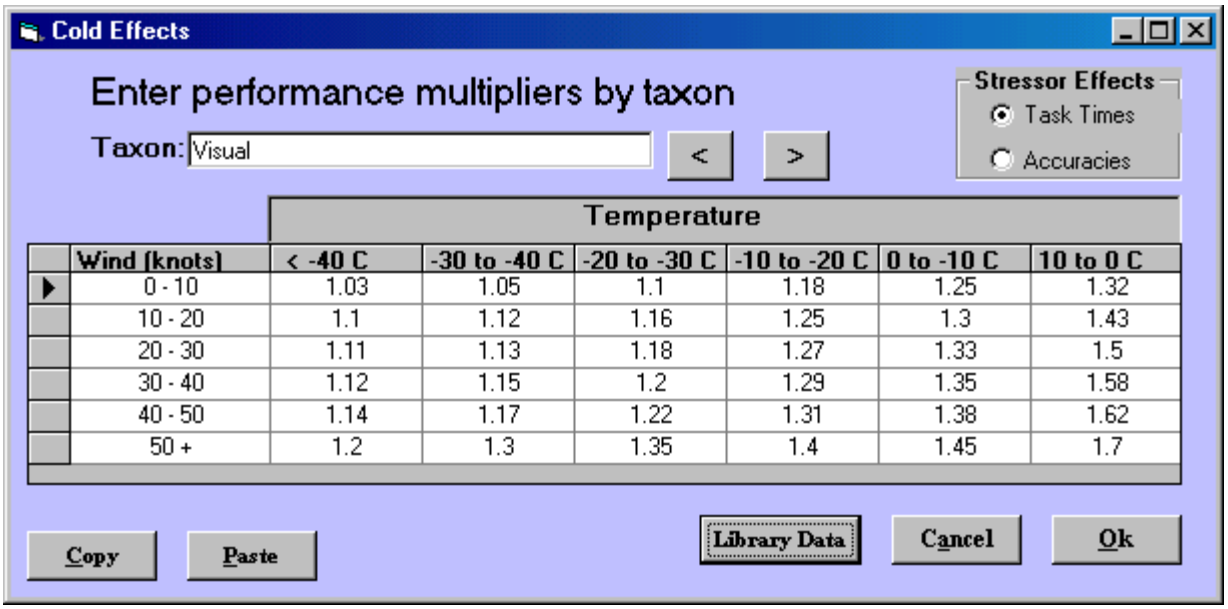

Figure 13. TESTIM interface to edit the cold stressor look-up table.

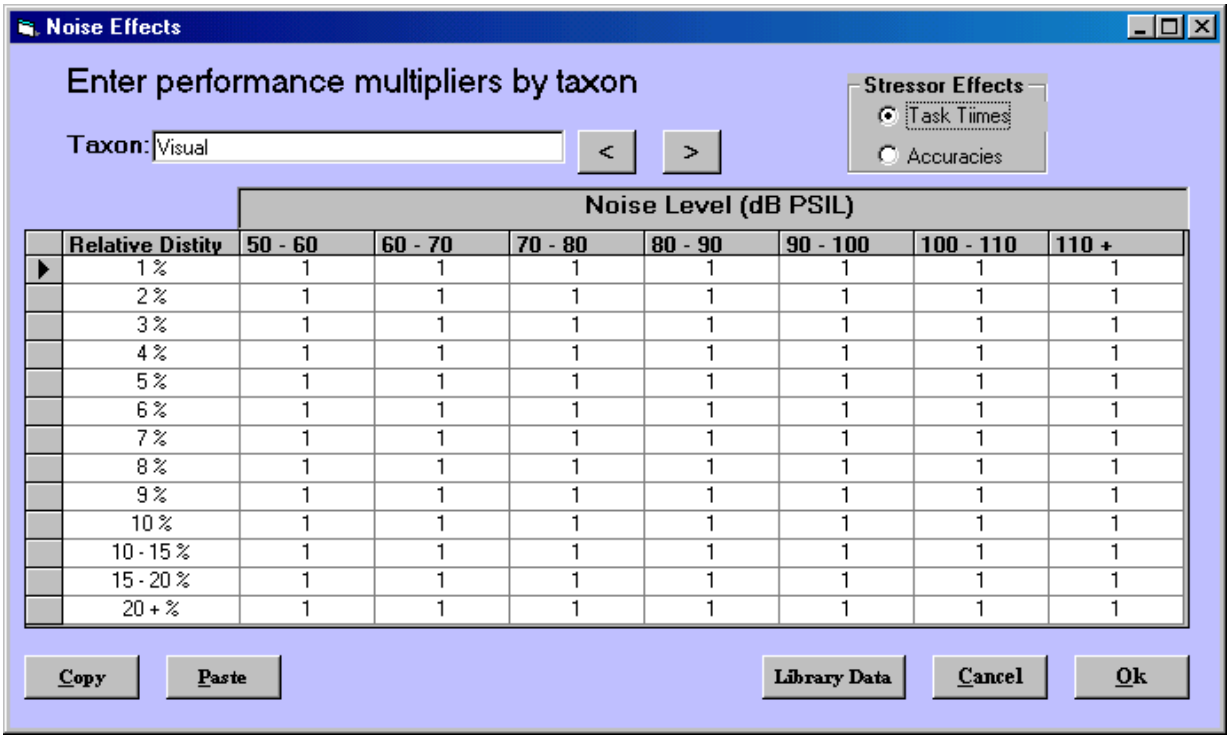

Figure 14. TESTIM interface to edit the noise stressor look-up table.

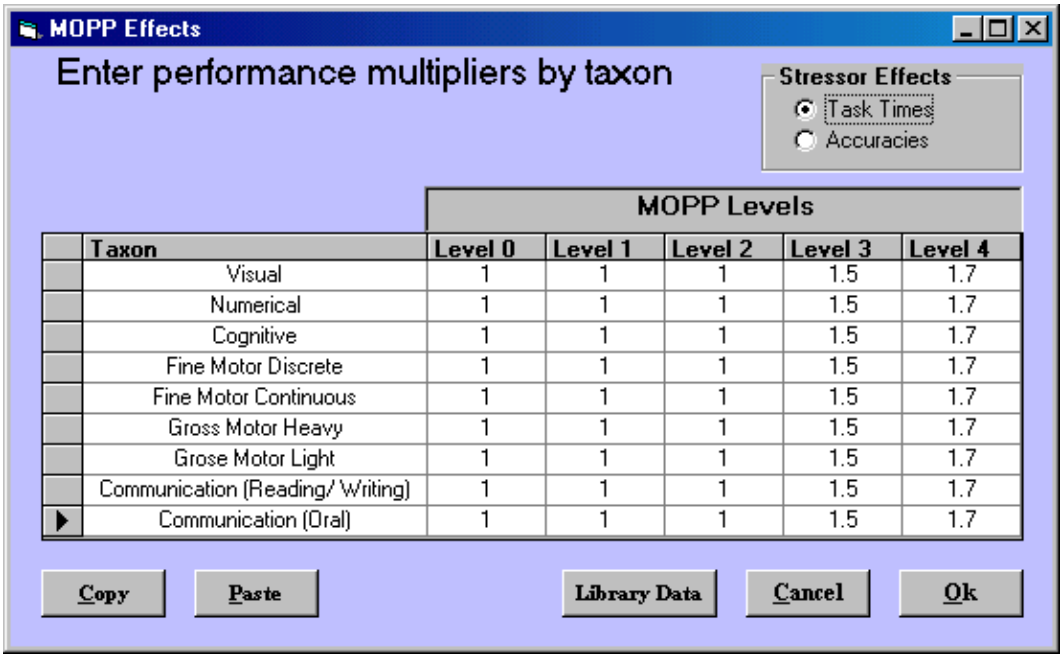

Figure 15. TESTIM interface to edit the MOPP stressor look-up table.

The "Experience" option of the Edit menu allows users to moderate the performance of task times and accuracies. The experience value is a simple multiplier that increases the task times for low experience and decreases the task times for high experience. The experience modifier increases the accuracy for high experience and decreases the accuracy for low experience. Figure 16 shows the experience moderator interface.

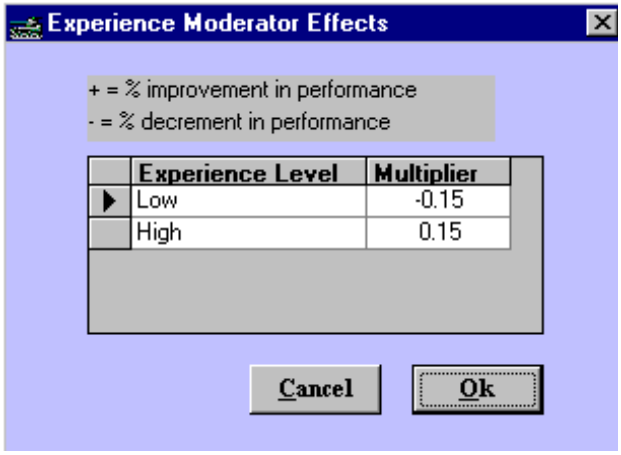

Figure 16. TESTIM interface to edit the experience moderators.

Map tasks menu. This menu contains options that allow users to map tasks in Performance Server models to the training and stressor taxonomies. When a user selects the "To Training Taxons" option, the application automatically scans the Performance Server model for any new tasks. If there are new tasks these will be added to the grid on the form "Map Tasks to

Training Taxons" shown in Figure 17. The user then weights each task based on the percentage of the task that represents each taxon.

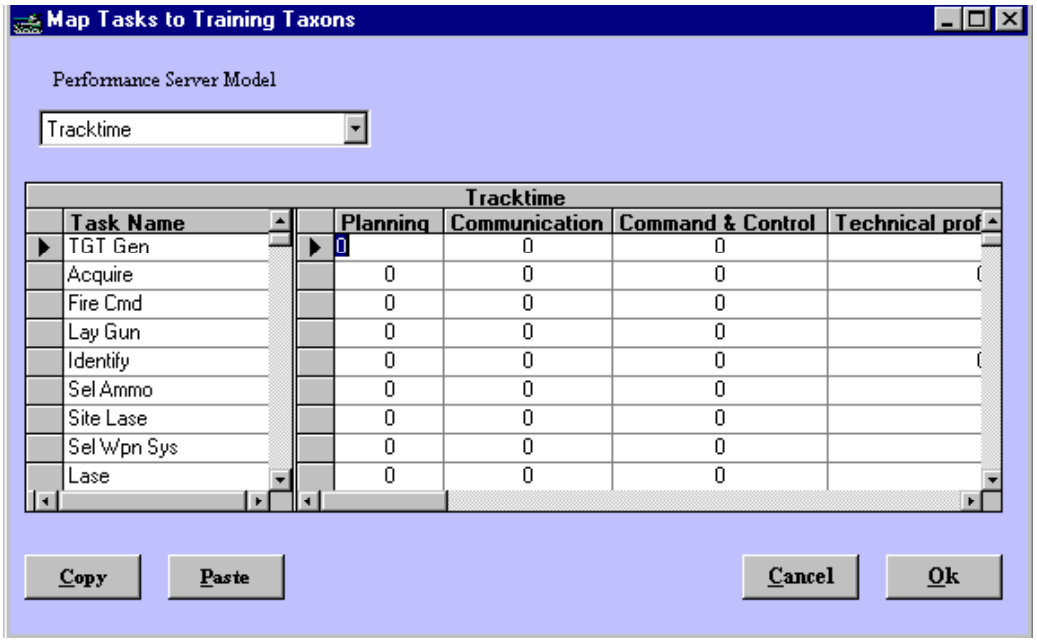

Figure 17. TESTIM interface mapping Performance Server model tasks to the training taxons.

The "To Stressor Taxons" option on the Map Tasks menu allows users to weight each task to the stressor taxons in the same manner that they were weighted for training taxons. Figure 18 shows the interface for mapping the tasks to stressor taxons.

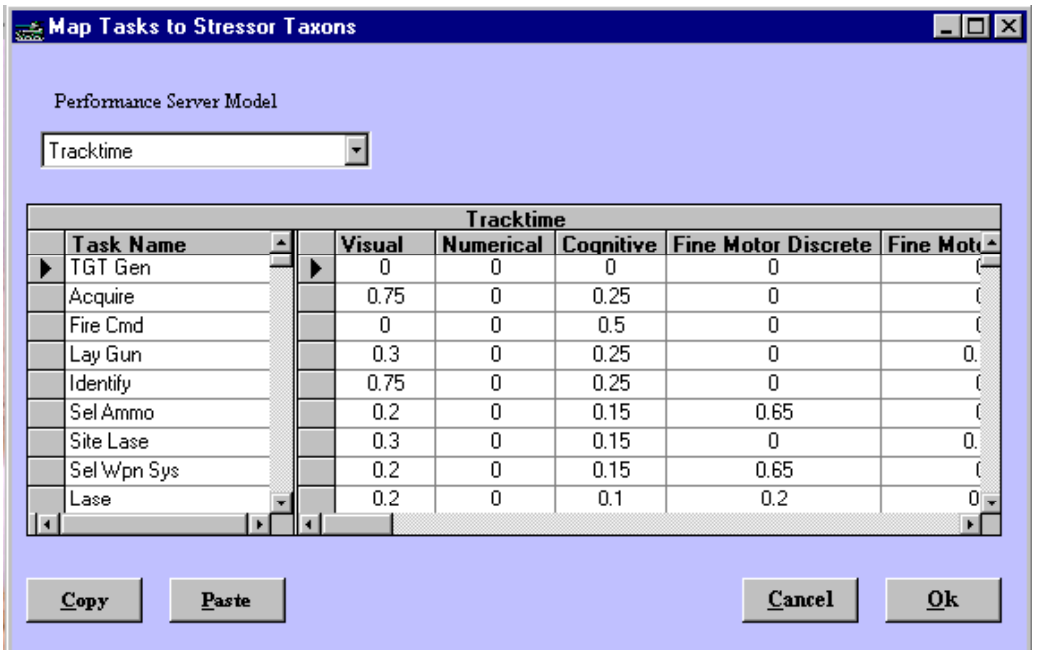

Figure 18. TESTIM interface mapping Performance Server model tasks to stressor taxons.

Apply menu. The "Make Default" option of the Apply menu causes the currently selected performance effects set to be the one that is used for all of the performance effects calculations for the Performance Server. When a user selects an effects set and then selects this option, TESTIM creates an OTB reader (RDR) file containing the names of the training types and taxons. The user must then directly copy this RDR file to a directory where it can be read by the OTB simulation. The OTB simulation will read this file and display the appropriate names in the OTB performance effects editor to obtain input from the OTB user.

Help menu. The "About TESTIM" option presents information on the version and date of the TESTIM software.

OTB Interfaces

This section describes the interfaces that were added to OTB for obtaining the necessary TESTIM input to calculate effects that are used in the Performance Server models. All of these interfaces are accessed from the Server Editor in OTB. A performance effects icon located on the tool bar along the top of the OTB display bring ups the Server Editor shown in Figure 19.

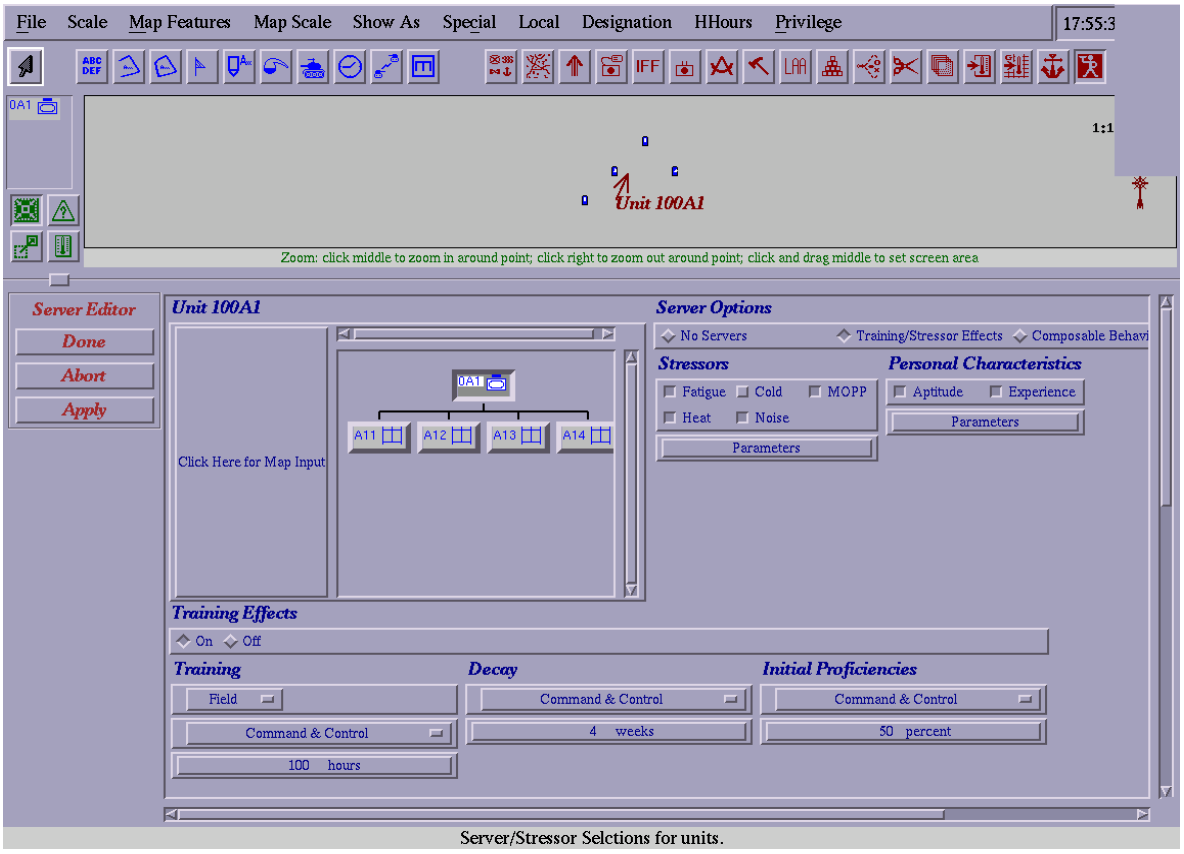

Figure 19. OTB interface for the performance effects editor.

The interface for the server editor contains two sections: Unit and Server Options. In the Unit section, a tree diagram indicating the unit structure is shown. This tree allows a user to specify which entities are to be affected by TESTIM. For example, Figure 19 shows a tank

platoon. If the top node labeled 0A1 is selected when the user presses the "Apply" button, then the effects will be applied to all of the tanks in that platoon. If one of the lower nodes on the tree (i.e., an individual tank) is selected when the user presses the "Apply" button, then only that tank will receive the effects. In order for an OTB simulation to be affected by TESTIM, the option labeled "Training/Stressor Effects" must be checked under the Server Options section. When the "Training/Stressor Effects" server is checked three new sections appear in the editor: Stressors, Personnel Characteristics and Training Effects.

Stressors. The "Stressors" section allows users to specify which stressor effects to use during the simulation by clicking the check box next to each stressor. More than one stressor can be selected. Some of the stressors have parameters that must be defined in order for the stressor algorithms to work correctly. Pressing the "Parameters" button next to a stressor produces another interface containing the appropriate input fields.

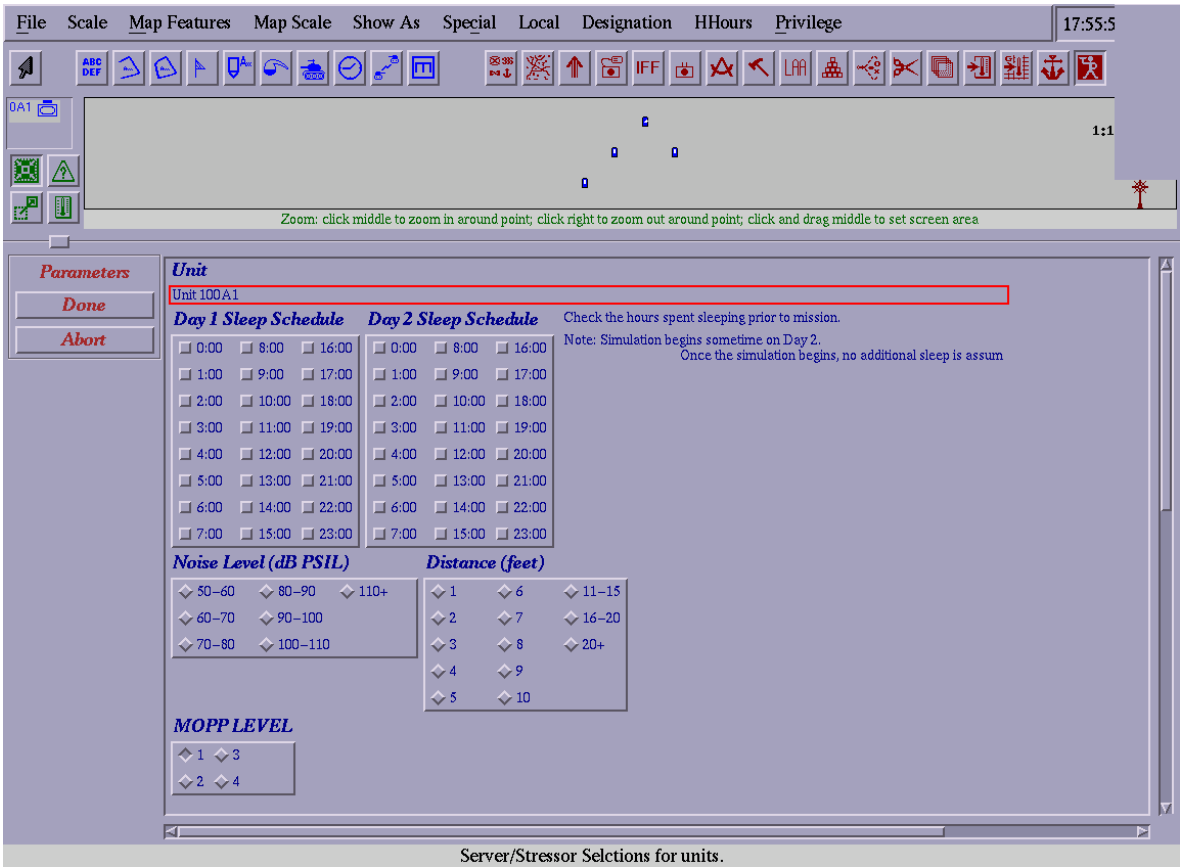

Figure 20. OTB interface for entering stressor parameters.

Depending on which stressors are checked in the performance effects editor, the user will have to enter different parameters using the interface shown in Figure 20. For fatigue, the user must specify the sleep schedule prior to the beginning of the battle exercise simulation. It is assumed that the exercise will start sometime on the second day. For noise, the input parameters are volume (in decibels - Preferred Speech Interference Level (dB PSIL)) and distance from the noise source (in feet). For MOPP, the user must select the level of MOPP gear that is being

worn by the soldier. The heat and cold stressor functions use variables from the OTB environmental model. The heat stressor function uses the air temperature and relative humidity as input parameters. The cold function uses air temperature and wind speed as input parameters. These input parameters are available on a moment-to-moment basis from the OTB environmental model. If the heat or cold stressors are checked on the performance effects editor interface and the environmental model in OTB is not turned on, the stressors will have no effect on performance. The OTB environmental model is turned on using the command line argument "-allowenv" when OTB is initially loaded.

Personnel characteristics. The section labeled "Personnel Characteristics" allows users to select aptitude and experience performance modifiers. The "Parameters" button allows users to enter parameters for both the aptitude and experience functions. Figure 21 shows the parameters interface for aptitude and experience. Users can define aptitude in one of two ways. They can either select an MOS or they can select an ASVAB cutoff score.

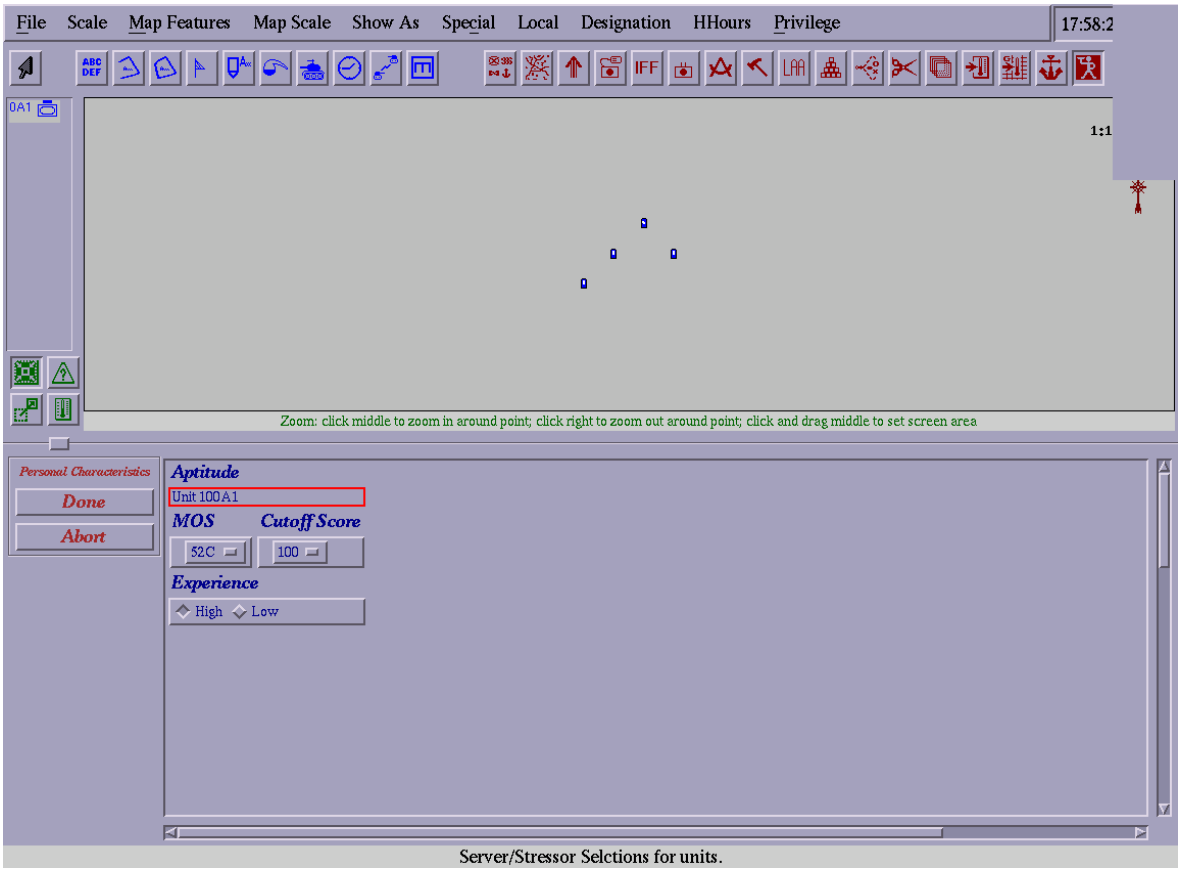

Figure 21. OTB interface for editing aptitude and experience parameters.

Training effects. The " Training Effects" section allows users to specify how much training an entity received for each training type and taxon pair. The user also specifies the number of weeks since that training occurred and the initial proficiency for each taxon just prior to the training. The "On" and "Off" buttons allow the user to specify whether to use the training

effects during the simulation. Figure 22 shows the interface that allows users to enter the number of training hours.

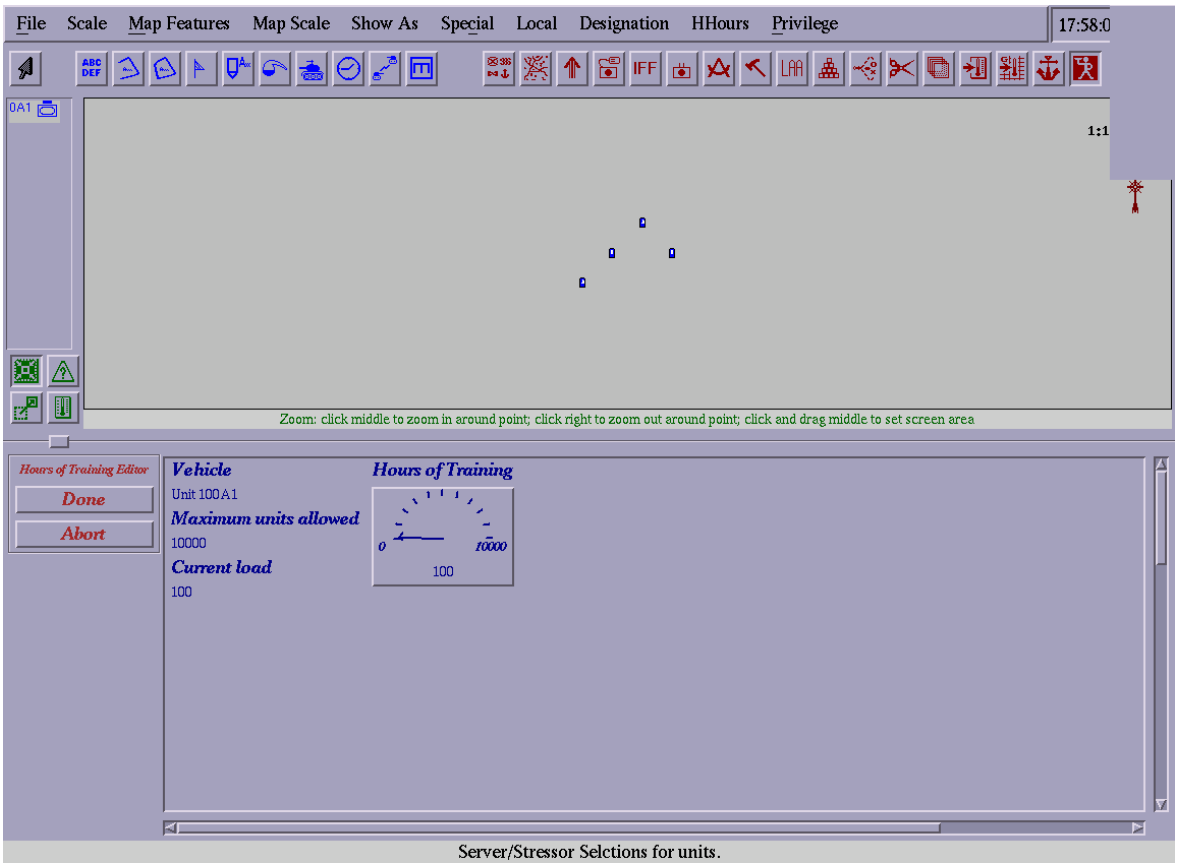

Figure 22. OTB interface for entering training hours.

Communication Architecture

The final major focus for Phase II of this SBIR was to develop the architecture for communicating the performance variable values between TESTIM and OTB. This architecture is illustrated in Figure 23 and includes three separate parts: the Performance Server, the middleware, and OTB. The Performance Server contains TESTIM and the human performance models of the entity behaviors that we are trying to affect in OTB. As described earlier, OTB contains interfaces that allow users to enter values that will be sent to TESTIM. The Performance Server is on a separate computer from the machine that will be running OTB. In between the two applications, there is middleware that manages the flow of data and keeps track of which entity needs what kind of performance value. The middleware communicates with the Performance Server via Microsoft's Component Object Model (COM) protocols and with OTB via DIS/HLA.

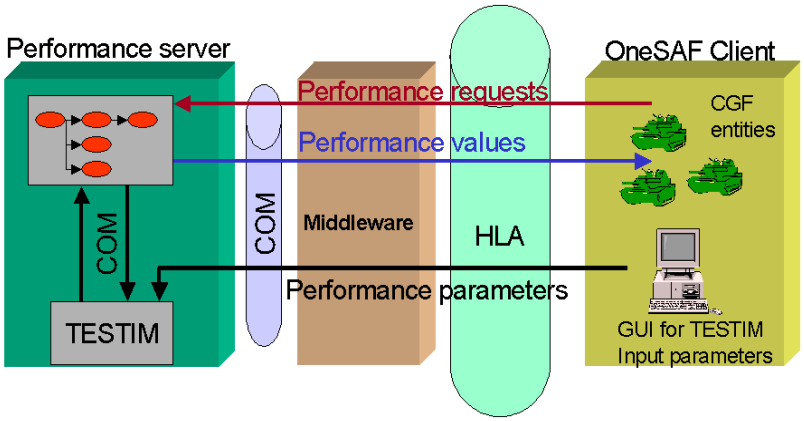

Figure 23. Communication architecture integrating TESTIM and OTB.

At the start of a simulation, input parameters that were entered by the OTB user are passed over the network to TESTIM. Then, as the OTB scenario executes, performance requests relating to various behaviors are passed to the Performance Server. When the Performance Server receives a request for a value from OTB, the model indicated within the request will begin executing. As each task in the model executes, a value will be sent to TESTIM indicating the need for a performance effect calculation. The performance calculations will be modifying the baseline task mean time values that have been entered into the models during model development. Variation in the actual task times in the model will come from drawing a distribution with the performance affected mean and the standard deviation. Variation in model branching logic (to represent decision making) based on proficiency values was also built into the models. The aggregated performance values are then sent back to OTB.

Using this approach, we can calculate the performance for OTB behaviors based on the training and stressor input values with very little modification of the OTB code. However, one of the challenges in developing this architecture was identifying the performance variables (called hooks) in OTB that can be modified. The hooks that were identified include: assault time, load time, subsequent load time, track time, subsequent track time, and the decision for determining whether to occupy an alternate position. For each of these hooks, submodels were developed using the simulation software *Micro Saint* and included in the Performance Server. Figure 24 shows the submodel for subsequent track time.

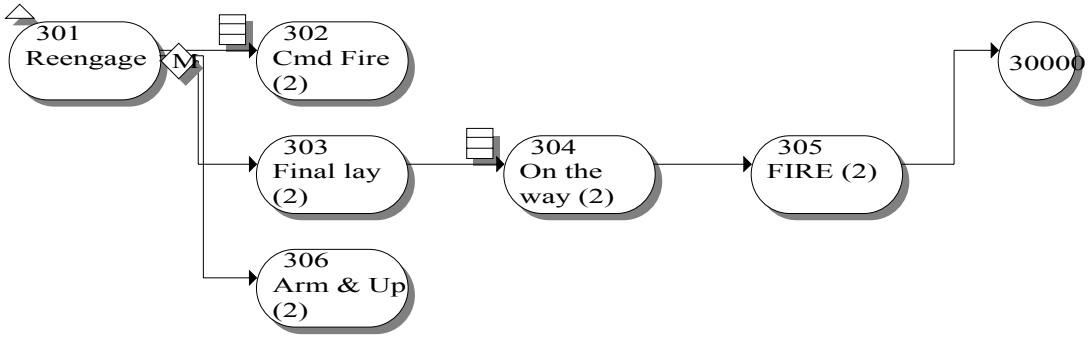

Figure 24. *Micro Saint* submodel to calculate subsequent track time.

Sensitivity Analysis

In order for the training and stressor algorithms to be useful and accepted by the user community, they need to accurately predict the effects of varying levels of training and stressors on performance. One method for evaluating these algorithms is validation. The Institute of Electrical and Electronics Engineers defines validation as "confirmation by examination and provisions of objective evidence that the particular requirements for a specific intended use are fulfilled."

A variety of objective techniques are available for validation. These include sensitivity analysis, consistency checks, and Turing tests. A sensitivity analysis is performed by systematically changing the values of model input variables and parameters over some range of interest and observing the effect on model behavior. As model input is changed, the output should change in predictable directions. Unexpected effects may reveal invalidity. The input values can also be changed to induce errors to determine the sensitivity of model behavior to such errors. A sensitivity analysis can identify those input variables and parameters to the values of which model behavior is very sensitive. Then, model validity can be enhanced by assuring that those values are specified with sufficient accuracy. In this study, a sensitivity analysis was conducted on our training algorithms using OTB. This analysis also allowed us to test the reliability of the TESTIM software, the performance editor interfaces in OTB, and the middleware. Due to time constraints, a sensitivity analysis was not performed on the stressor algorithms.

OTB Simulation

Figure 25 shows the OTB simulation developed for the sensitivity analysis. The simulation contained an M1 tank platoon for the friendly forces and a T-72 M tank platoon for the enemy forces over the Fort Knox terrain database. The mission for the friendly forces was to perform a hasty occupy position and the mission for the enemy forces was to perform an assault on the friendly forces. Rules of engagement for both forces were set to free. All other parameters in the simulation were kept at OTB default settings.

Test Plan

The default library within TESTIM consists of three training types (classroom, field, and simulator), five taxons for which soldiers can receive training (communication, command and control, planning, tactics and doctrine, and technical proficiency with equipment), decay rates for each taxon, five stressors (cold, fatigue, heat, MOPP, and noise), two experience levels, and multiple levels of aptitude. In order to analyze TESTIM's algorithms completely, over 3000 combinations of these variables would need to be entered into the OTB test model. Rather than perform such an extensive and time-consuming test, a reduced test plan was conducted to show the positive correlation between training and performance (i.e., as training increases, performance increases).

The independent variable was the proficiency for the friendly M1 tanks. This proficiency was the same for each taxon within a particular simulation run. The values tested were 10, 20,

30, 40, 50, 60, 70, 80 and 90 percent proficiency. Again, the effects of time since training, aptitude, stressors, and experience were turned off. Three dependent variables were measured: mission success, load time, and subsequent track time. A successful mission was defined as one in which all of the enemy tanks were destroyed (fire power or catastrophic kill) or the enemy tanks retreated. Note that even under exactly the same environmental and mission parameters, the outcome of a mission is not always the same due to the variability within OTB. Therefore, the OTB simulation was run ten times for each proficiency value to obtain a sufficient number of mission outcomes. Because loading and tracking occurs throughout a simulation, the performance times for these tasks were only collected in one out of the ten simulation runs to reduce the amount of data collected.

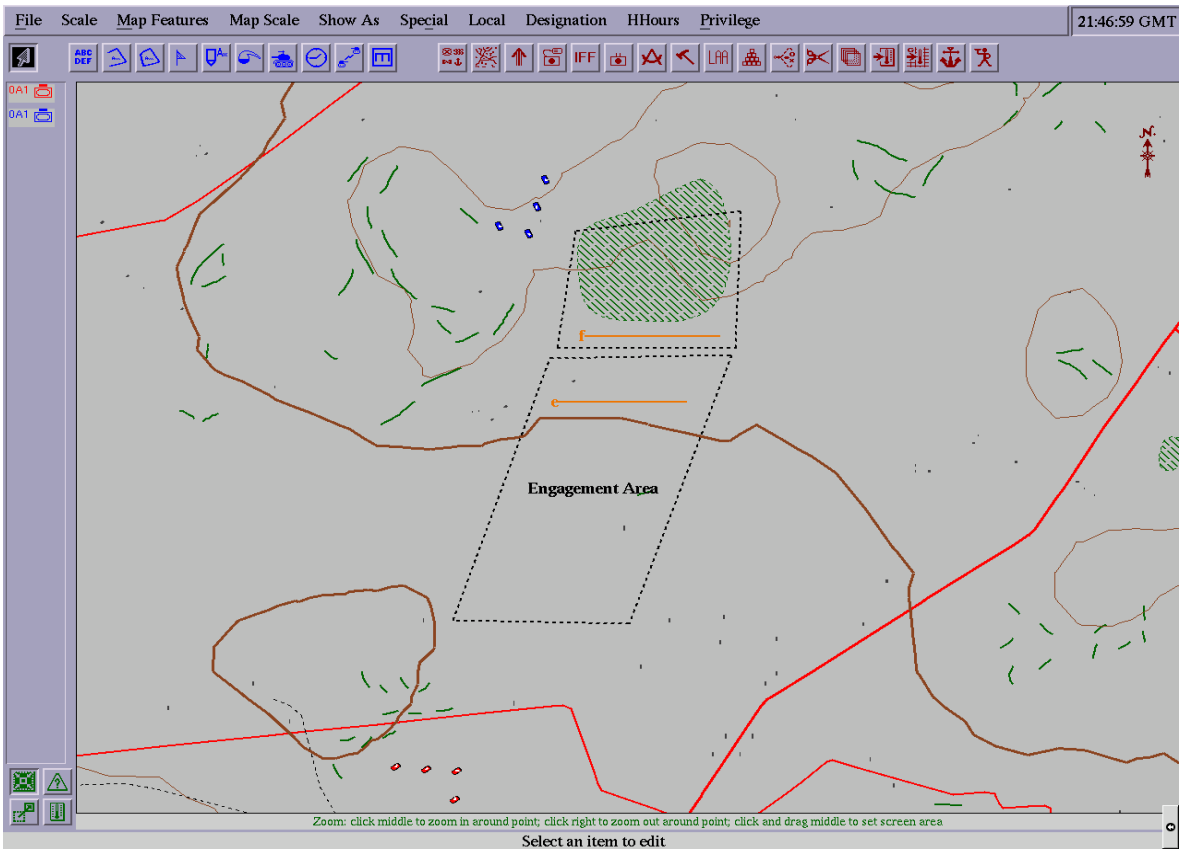

Figure 25. OTB simulation used for the sensitivity analysis.

Before the sensitivity analysis was started, we used the learning and decay functions to determine how a soldier's proficiency changed based on different amounts of training. It is the soldier's proficiency that is used to adjust task times and accuracies within TESTIM. Table 1 shows the resulting proficiencies (in percentages) based on different amounts of training for each taxon. The starting proficiency (before training) for each taxon and training type was 10%.

Table 1. Resulting proficiencies based on training for each taxon.

	Hours of training			
Taxon / training type	10	20	30	40
Planning				
Classroom	50	51	51	51
Simulator	73	73	73	73
Field	76	82	85	87
Communication				
Classroom	35	39	41	42
Simulator	70	70	70	70
Field	76	82	85	86
Command and control				
Classroom	47	50	51	51
Simulator	75	78	79	79
Field	58	65	69	72
Technical proficiency with equipment				
Classroom	24	27	29	30
Simulator	38	43	45	47
Field	64	71	75	77
Tactics and doctrine				
Classroom	39	41	42	42
Simulator	33	38	40	42
Field	58	65	69	71

Figure 26 graphically illustrates the effects of different amounts of field training on a soldier's proficiency for the planning taxon. The combined effects of zero and ten hours of classroom (C) and simulator (S) training are also included. Table 2 shows the resulting proficiencies (in percentages) based on different amounts of decay for each taxon. The starting proficiency (after training) for each taxon and training type was 90%.

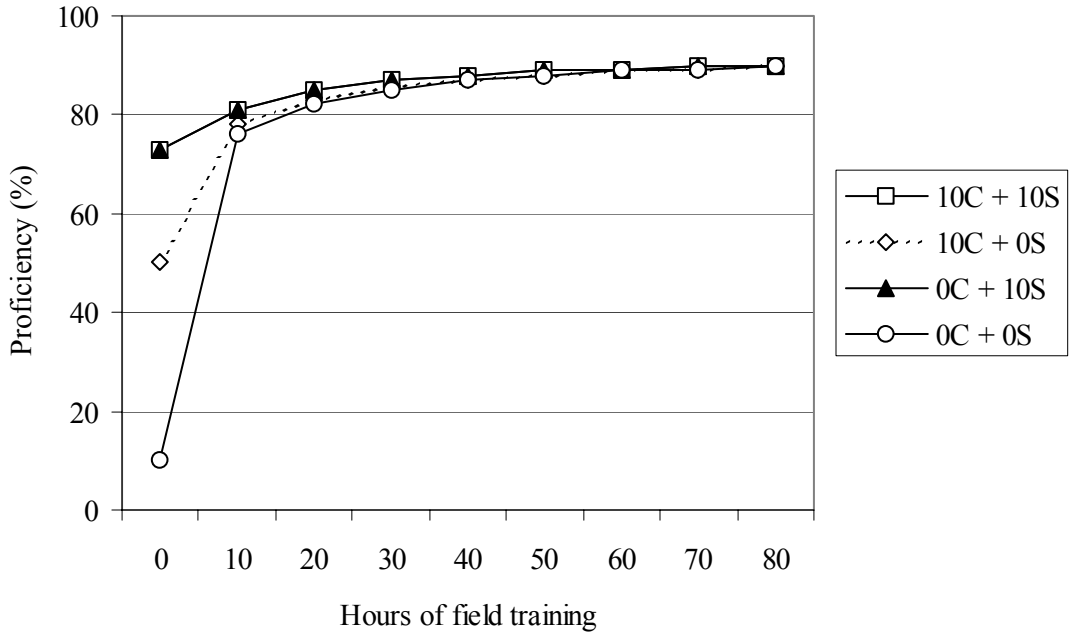

Figure 26. The effects of training for the planning taxon.

Table 2. Resulting proficiencies based on decay for each taxon.

	Time since training (months)			
Taxon	6	12	18	24
Planning	77	73	70	69
Communication	75	70	67	65
Command and control	76	71	68	65
Technical proficiency with equipment	76	72	69	68
Tactics and doctrine	85	83	82	81

Results

Using an analysis of variance (ANOVA), a significant main effect was found for proficiency for both load time and subsequent track time [$F(8,208) = 616.40$, $p < 0.01$ and $F(8,225) = 4920.64$, $p < 0.01$, respectively]. As proficiency increased, the average load and subsequent track times decreased (see Figure 27). A post hoc analysis, the Tukey Honest Significant Difference (HSD) test, was used to determine which proficiencies produced significantly different results. Table 3 and Table 4 show that most of the differences were significant. No significant effect was found for proficiency and mission success.

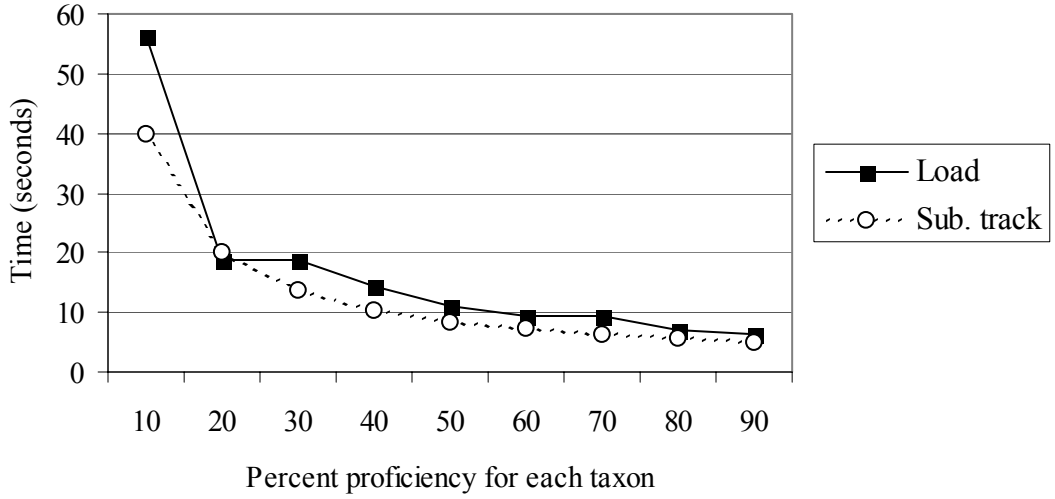

Figure 27. Load and subsequent track times based on an entity's proficiency for each taxon.

Table 3. Results of the Tukey HSD test for load time.

	Proficiency								
Proficiency	10	20	30	40	50	60	70	80	90
10		*	*	*	*	*	*	*	*
20	*			*	*	*	*	*	*
30	*			*	*	*	*	*	*
40	*	*	*		*	*	*	*	*
50	*	*	*	*				*	*
60	*	*	*	*				*	*
70	*	*	*	*					
80	*	*	*	*	*	*			
90	*	*	*	*	*	*			

* Comparison was significantly different, $\alpha_{FW} = .01$.

Table 4. Results of the Tukey HSD test for subsequent track time.

	Proficiency								
Proficiency	10	20	30	40	50	60	70	80	90
10		*	*	*	*	*	*	*	*
20	*		*	*	*	*	*	*	*
30	*	*		*	*	*	*	*	*
40	*	*	*		*	*	*	*	*
50	*	*	*	*		*	*	*	*
60	*	*	*	*	*		*	*	*
70	*	*	*	*	*	*			*
80	*	*	*	*	*	*			
90	*	*	*	*	*	*	*		

* Comparison was significantly different, $\alpha_{FW} = .01$.

The sensitivity analysis showed that, as expected, TESTIM's algorithms have a significant effect on task performance within OTB. The lack of any effect of training on overall mission success can be explained in two ways. First, the M1 tanks used for the friendly forces are superior to the T-72 M tanks used for the enemy forces. Typically, the enemy forces would use all of their ammunition early in a simulation. By the time the enemy tanks reached the battle position, they had nothing left to fire and were destroyed by the friendly tanks - even when the friendly forces took a significantly long time to perform loading and tracking tasks. Second, only a few of the several hundred tasks within OTB are currently affected by the TESTIM algorithms. We believe that once additional tasks and decisions within OTB are affected by TESTIM, the larger the impact of training and stressors will have on overall performance.

Conclusions

In this project, we feel that we have developed an innovative way to link training and stressors to performance. We have collected some empirical data from soldiers to anchor and populate the algorithms. We have developed an authoring tool for developing custom training and stressor effects. We have implemented a robust and scalable approach for linking human performance calculations to constructive simulations.

The algorithms, software, and Performance Server architecture for improving the realism of CGF have been developed and demonstrated. However, only a very small number of CGF behaviors have been addressed with this approach to date. Countless more need to be addressed before the claim to have truly 'improved the realism of CGF' can be made. Behaviors need to be identified in OTB and models need to be developed to address those behaviors. Human performance data also need to be collected to make the models valid. Better, quantifiable training effects data are especially needed.

A number of opportunities for synergy between the HSI and training communities and the virtual simulation developers can help to accommodate these needs. The HSI community has been developing human performance models of military operations for several years. Many of these models include the same behaviors that need to be improved in CGF. Many of these models have also been validated. If an extensive list of behaviors from simulations like OTB were developed, the wealth of HSI models could be used as a starting point for Performance Server models rather than starting from scratch.

A second opportunity for synergy is that most of the HSI models have been designed and developed to help analysts identify 'high driver' behaviors. That is; behaviors that are critical determinants of total system performance. Training personnel and commanders who develop OTB scenarios to train soldiers can use this information to be sure that the scenarios include practice using these kind of tasks and scenarios.

To address the desperate need for quantifiable data on how training affects performance, data could be collected from the man-in-the-loop simulators to record how the human participants perform on critical tasks. If data were also collected on the amount and type of training each participant has had prior to the simulation exercise it could eventually be determined how training affects performance. This would be a long and arduous study but it would be very non-intrusive to the soldiers and the combat units. The result of exploiting this kind of opportunity for synergy would be an expanding empirical data set to further our understanding of the impact of training on performance.

A fourth opportunity for synergy also exploits man-in-the-loop simulation as a data collection opportunity. By collecting data on exactly what the man-in-the-loop does during the simulation, network models of task performance could be extracted from the data for building additional Performance Server models. This information could also be passed on to the HSI community to help improve their models and build new ones, ultimately supporting the use of quantitative analyses to support system acquisition.

The work described in this report represents a way for human performance professionals to get involved in the development of virtual simulations without interfering with the development or with the training community's ability to achieve their goals of providing cost-effective training opportunities. It also describes several potential ways that the training, HSI, and simulation communities can work together to share the products of their work.

Follow-On Recommendations

The work performed for this effort represents a first step in incorporating research that has been done on the effects of training and performance shaping factors into algorithms and data structures that can be applied to a variety of human performance modeling environments. However, it is only a first step. Different research programs around the world are collecting data to understand the complex *interactions* of humans with factors that influence their performance. Based on this research, there is the potential that some of the algorithms within TESTIM will need to be modified to allow human performance modeling to more accurately predict the way that humans actually behave under a variety of conditions.

A second opportunity for follow-on work is to identify additional hooks and vehicles in OTB to be modified by TESTIM. Currently, TESTIM affects the task times for assault, load, track, subsequent load, and subsequent track in OTB. TESTIM also affects the decision for moving to an alternate position during a hasty occupy position. Allowing TESTIM to affect other hooks in OTB would increase the realism of the OTB simulations. The following is a list of potential hooks we might affect:

1. Vehicle level hooks
    a. Change the amount of time an entity spends in its hidden position during an occupy position task.
    b. Change how close indirect fire needs to land next to an entity before the entity will consider it accurate fire

2. Unit level hooks
    a. Change what percentage of the assaulting unit must be destroyed before stopping the assault
    b. Change how soon a unit reassesses its enemy situation after reacting to enemy contact by deploying smoke
    c. Modify a unit's reaction to enemy contact
    d. Modify the distance that a unit will advance beyond the objective to secure a defendable position
    e. Modify the width of the defensive position assumed after the assault.
    f. Change the distance between an entity's primary and alternate firing positions
    g. Change when a unit determines if it is blocked by smoke
    h. Change the amount of smoke coverage an entity uses during a withdraw

A third opportunity for follow-on work is to integrate decision-making models with TESTIM and include more robust branching logic to represent human decision-making. At those decision points where an entity must decide what to do, we can represent the underlying decision making process which, in turn, can be conditioned by a variety of factors, rather than depending solely on irreducible probabilities that dictate what the entity decides to do. In this context, the underlying human performance model can be as simple as a handful of production rules that define what the entity will do given the current state of the environment. Alternatively, we might embed a more nuanced model of the human decision-making process. For instance, we could include a "naturalistic" model of decision-making currently under development at MA&D. In place of rigid production rules, this model makes use of a fuzzy recognition routine together with a model of the decision-maker's long-term memory to determine the appropriate course of action. Thus, decisions are made on the basis of the agent's experience and, moreover, can be affected by both changes in that experience and by the effects of stress and uncertainty.

Finally, there is the potential to incorporate additional functionality to the Performance Server. This includes the following:

- Using HLA instead of DIS as the network protocol.
    - HLA is more stable than DIS.

- Allowing for dynamic subscription.
  - This allows for the possibility of decreasing processor load, thus requiring fewer computers to run a simulation.
- Allowing for the push of information rather than the pull of information.

References

Abel, M. H. (1986). Performance of soldiers on the battlesight tank gunnery video game. (Report No. ARI-TR-710). Alexandria, VA: Army Research Institute for the Behavior and Social Sciences.

Aoki, M. (1971). Introduction to optimization techniques: Fundamentals and applications of nonlinear programming. New York: Macmillian.

Archer, R., Walters, B., Yow, A., Carolan, T., & Laughery, K. R. (1999). Improving soldier factors in prediction models. (Phase I Final Rep.). Alexandria, VA: Army Research Institute for the Behavior and Social Sciences.

Bahrick, H. P., & Hall, L. K. (1991). Lifetime maintenance of high school mathematics content. Journal of Experimental Psychology: General, 104, 54-75.

Blockley, W. V., & Lyman, J. (1951). Studies of human tolerance for extreme heat IV: Psychomotor performance of pilots as indicated by a task simulating aircraft instrument flight. (Report No. 6521). Dayton, OH: Wright-Patterson Air Force Base.

Bradley, C. M., & Robertson, K. A. (1998). Combined stressors and performance: A review. Hampshire, UK: Defence Evaluation and Research Agency.

Burden, R. L., & Faires, J. D. (1997). Numerical analysis (6^{th} ed.). Pacific Grove, CA: Brooks/Cole Publishing Company.

Cannon-Bowers, J. A., & Salas, E. (1998). Making decisions under stress: Implications for individual and team training. Washington, DC: American Psychological Association.

Engh, T., Yow, A., & Walters, B. (1998). An evaluation of discrete event simulation for use in operator and crew performance simulation (Final Rep.). Washington, DC: U.S. Nuclear Regulatory Commission.

Fitts, P. M. (1962). Factors in complex skill training. In Glaser, R. (Ed.) Training research and education. New York: Wiley.

Gillis, P. D., & Hursh, S. (1999). Using behavior modifiers to influence CGF command entity effectiveness and performance. Proceedings of the Eighth Conference on Computer Generated Forces and Behavioral Representation, Orlando, FL.

Glasow, P., & Pace, D. R. (1999, January). Making VV&A effective and affordable (Mini-Symposium Report). SIMVAL99, Laurel, MD.

Gottfredson, L. S. (1997). Why g matters: The complexity of everyday life. Intelligence, 24, 79 - 133.

Graham, S. E., & Smith, T. L. (1990). Identifying tank gunnery skill requirements on the institutional conduct-of-fire trainer (I-COFT). (Report No. ARI-RR-1583). Oklahoma City, OK: Oklahoma University Foundation Incorporated.

Hancock, P. A. (1986). The effect of skill on performance under an environmental stressor. Aviation, Space and Environmental Medicine, 57 (1), 59-64.

Hancock, W. M., & Bayha, F. H. (1992). The learning curve. In G. Salvendy (Ed.), Handbook of industrial engineering (pp. 1585-1598). New York: Wiley.

Harris, D. (1985). A degradation methodology for maintenance tasks. HQDA, MILPERCEN (DAPC-OPA-E). Alexandria, VA.

Hart, R. J., Hagman, J. D., & Bowne, D. S. (1990). Tank gunnery: Transfer of training from TopGun to the Conduct-of-Fire Trainer. (Report No. ARI-RR 1560). Alexandria, VA: Army Research Institute for the Behavior and Social Sciences.

Hoffman, R. G., & Melching, W. H. (1984). Field trials of the MK60 tank gunnery simulator in armor institutional training courses. (Report No. HUMRRO-FR-TRD(KY)-82-9-vol-1). Alexandria, VA: Human Resources Research Organization.

Kraemer, R. E., & Smith, S. E. (1990). Soldier performance using a part-task gunnery device (TOPGUN) and its effects on Institutional-Conduct of Fire Trainer (I-COFT) proficiency. (Report No. ARI-RR-1570). Alexandria, VA: Army Research Institute for the Behavior and Social Sciences.

Lane, N. E. (1986). Skill acquisition curves and military training. (Report No. IDA-P-1945). Alexandria, VA: Institute for Defense Analysis.

Lowry, J. C., Rappold, V. A., & Copenhaver, M. M. (1992). Feasibility study for predicting human reliability growth through training and practice. (Report No. ARI-RN-92-39). Alexandria, VA: Allen Corp of America.

Luh, C. W. (1922). The conditions of retention. Psychological Monographs, Whole No. 142, Vol. XXXI, No. 3.

Mackworth, N. H. (1950). Researches on the measurement of human performance. London: HMSO.

McHenry, J.J., Hough, L. M., Toquam, J. L., Hanson, M. A., & Ashworth, S. (1990). Project A validity results: The relationship between predictor and criterion domains. Personnel Psychology, Special Issue on the Army Selection and Classification Project (Project A), 43 (2), 335-355.

Micro Analysis and Design Incorporated (1999). [IPME specification information]. Boulder, CO: Dave Dahn. Unpublished specification.

Micro Analysis and Design Incorporated, & Dynamics Research Corporation (1999). Enhanced performance degradation factors and upgrades for Improved Performance Research Integration Tool (IMPRINT) Version 5: Stressor review report. (Contract DAAL01-95-C-0122). Boulder, CO: Susan Archer, Rich Adkins, & Brett Walters.

Morrison, J.E., Drucker, E. H., & Campshure, D. A. (1991). Devices and aids for training M1 tank gunnery in the Army National Guard: A review of military documents and the research literature. (Report No. HUMRRO-IR-PRD-90-19). Alexandria, VA: Human Resources Research Organization.

Newell, A., & Rosenbloom, P. S. (1981). Mechanisms of skill acquisition and the law of practice. In J. R. Anderson (Ed.), Cognitive skills and their acquisition. Hillsdale, NJ: Erlbaum.

Nichols, P. D., Chipman, S. F., & Brennan, R. L. (Eds.). (1995). Cognitively diagnostic assessment. Hillsdale, NJ: Erlbaum.

Pew, R. W., & Mavor, A. S. (Eds.). (1998). Human behavior representation: Military requirements and current models. In Modeling human and organization behavior: Application to military simulations (pp. 19-50). Washington, DC: National Academy Press.

Rose, A. M. (1989). Acquisition and retention of skills. In G. Macmillan (Ed.), Applications of human performance models to system design. New York: Plenum Press.

Rose, A. M., Czarnolewski, M. Y., Gragg, F. E., Austin. S. H., Ford, P., Doyle, J., & Hagman, J. D. (1985). Acquisition and retention of soldiering skills. (Report No. ARI-TR-671). Alexandria, VA: Army Research Institute for the Behavior and Social Sciences.

Sterling, B. S. (1996). Relationship between platoon gunnery training and live-fire performance. (Report No. ARI-RR-1701). Alexandria, VA: Army Research Institute for the Behavior and Social Sciences.

Sticht, T. G., Armstrong, W. B., Hickey, D. T., & Caylor, J. S. (1987). Cast-off youth: Policy and training methods from the military experience. New York: Praeger.

Strang, G. (1988). Linear algebra and its applications (3rd ed.). Fort Worth, TX: Harcourt Brace Jovanovich College Publishers.

Turnage, J. J., & Bliss, J. P. (1989). Training transfer in a tank gunnery training system. Proceedings of the Human Factors Society 33rd Annual Meeting, 1315-1319.

Wackerly, D. D., Mendenhall, III, W., & Scheaffer, R. L. (2002). Linear models and estimation by least squares. In Mathematical statistics with applications (6th ed., Chapter 11). Belmont, CA: Duxbury.

Walters, B., French, J., & Barnes, M. J. (2000). Modeling the effects of crew size and crew fatigue on the control of Tactical Unmanned Aerial Vehicles (TUAVs). In J. A. Joines, R. R. Barton, K. Kang, & P. A. Fishwick (Eds.), Proceedings of the 2000 Winter Simulation Conference (pp.920-924). Piscataway, NJ: Institute of Electrical and Electronics Engineers.

Appendix A
Training Effects Data Collection Questionnaire

Micro Analysis and Design is working with the Army Research Institute to develop a usable model of training that can be embedded into Computer Generated Force (CGF) models. The primary goal of this effort is to develop algorithms and data structures for including the effects of training into these models. The tasks for which we are asking you to provide training estimates are the 38 top level tasks from the ARTEP 17-237-10 Mission Training Plans (MTP) for Tank Platoons. We need to collect data on *how much* training and *what kind* of training you received for each MTP task for each of the four previous quarters. For each task we also need to collect data on your initial ability and the maximum amount of benefit you feel that each type of training can provide you. The challenge of this effort is for us to be able to generalize the effects of several training variables across all of the tasks that soldiers can perform. We need help from subject matter expert soldiers to provide estimates of data that will serve as input into our model.

Please note that the information obtained from this questionnaire will in NO WAY be used as an evaluation mechanism for either you or your unit. We will not be collecting the names or the units of the soldiers that participate in this data collection effort. The sincerity of your responses will contribute to the accuracy of our training effects model. The following describes the process for completing the questionnaire and examples of hypothetical entries.

For each task:

1. Enter your innate proficiency in percentage. Your innate proficiency on a task is how well you performed the task the first time you performed it (i.e. after a minimal amount of training).
2. Enter asymptotes for each type of training. The asymptote is the highest level of proficiency you feel that you can get to after training with a particular type of training.
3. Enter your percent proficiency at the beginning of the last year.

For each quarter and each task:

4. Enter the number of hours of each type of training you received.
5. Enter your percent proficiency at the end of the quarter.

Repeat steps 3 through 5 for each quarter in the past year, starting with the beginning of the previous year. It may be helpful for you to obtain your training records for the last year.

<u>Example</u>

The following example illustrates a sample set of data from a soldier regarding his training on Platoon Battle Drills. The first portion of the survey involved the soldier entering in his innate proficiency for each task and asymptotes for each type of training for each task. For the first task (Change of Formation Drill), the soldier said that he believes that the first time he perform this task he did it at only 20% proficiency. He also believes that if he were only to have classroom training, no simulator or field, that he would only be able to reach a proficiency level

of 50%. This means that no matter how many hours of classroom training he receives he would never get any better at the task then 50% without simulator or field training. For this same task he believes that simulator training could get him to 90% proficiency and field training could get him all the way to 98% proficiency. This means that he feels that a simulator could train him such that he makes errors 10% of the time while performing the task, but only field training can make it so that he only makes errors 2% of the time.

In the second part of the sample survey the soldier indicates how much training he has had for each task during each quarter over the last year. In this example we only examine the soldiers first quarter of training on a few tasks. For the first task (Change of Formation Drill), the soldier received no training and he believed that his proficiency at this task remained the same from the beginning to the end of the quarter. For the second task (Contact Drill), the soldier received 20 hours of field training and 2 hours of classroom training. The soldier felt that his proficiency at the beginning of the quarter was 75% and it improved to 85% at the end of the quarter. In the third task (Action Drill), the soldier received 16 hours of field training, 4 hours of simulator training, and 1 hour of classroom training. He indicated that this training resulted in an increase in his ability to perform the task by 10%.

The questionnaire can also be used to describe and evaluate a decrease in proficiency for the quarter. In the example task 4 (React to Indirect Fire), the soldier believed he had a proficiency of 80% at the beginning of the quarter. However, due to a lack of training he felt that his proficiency had dropped to 70% by the end of the quarter. Please note that the values chosen for these examples are to illustrate the process for completing the questionnaire and are not meant as realistic training values.

Sample Questionnaire Data:
General Information for Each Task

ARTEP 17-237-10-MTP TANK PLATOON (1996/07/09)	Innate Proficiency	Maximum Proficiency for Field Training	Maximum Proficiency for Simulator Training	Maximum Proficiency for Classroom Training
BOS: PLATOON BATTLE DRILLS				
Change of Formation Drill (BD-1)	20	98	90	50
Contact Drill (BD-2)	30	95	95	50
Action Drill (BD-3)	20	95	80	60
React to Indirect Fire Drill (BD-4)	25	98	90	60
React to Air Attack Drill (BD-5)	15	90	90	40
React to a Nuclear Attack Drill (BD-6)	10	90	85	40
React to a Chemical/Biological Attack Drill (BD-7)	10	90	85	40

Information for Each Task for Quarter 1

ARTEP 17-237-10-MTP TANK PLATOON (1996/07/09)	Starting Proficiency	Hours of Field Training	Hours of Simulator Training	Hours of Classroom Training	Ending Proficiency
BOS: PLATOON BATTLE DRILLS					
Change of Formation Drill (BD-1)	80	0	0	0	80
Contact Drill (BD-2)	75	20	0	2	85
Action Drill (BD-3)	80	16	4	1	90
React to Indirect Fire Drill (BD-4)	80	0	0	0	70
React to Air Attack Drill (BD-5)	75	20	0	1	85
React to a Nuclear Attack Drill (BD-6)	60	10	0	4	80
React to a Chemical/Biological Attack Drill (BD-7)	60	10	0	4	80

DEFINITION: For the purpose of this questionnaire, we define 100% proficient as being able to perform the task without any errors 100% of the time.

TRAINING EFFECTS QUESTIONNAIRE: GENERAL

ARTEP 17-237-10-MTP TANK PLATOON (1996/07/09)	Innate Prof.	Max. Prof. for Field Training	Max. Prof. for Sim. Training	Max. Prof. for Class. Training
BOS: COMMAND AND CONTROL				
Conduct Troop-Leading Procedures (17-3-0065)				
Conduct Assembly Area Activities (17-3-2000)				
Conduct Linkup (17-3-2760)				
BOS: INTELLIGENCE				
Establish Observation Posts (17-3-1039)				
BOS: MANEUVER				
Conduct Bypass Operations (17-3-2420)				
Conduct Convoy Escort (17-3-2320)				
Coord/Conduct a Passage of Lines Forward/Rearward (17-3-1014)				
Conduct Tactical Movement (17-3-1016)				
Conduct a Tactical Road March (17-3-0212)				
Execute Actions on Contact (17-3-0221)				
Destroy an Inferior Force (17-3-2450)				
Assault an Enemy Position (17-3-0220)				
Conduct an Attack by Fire (17-3-0219)				
Conduct Overwatch/Support by Fire (17-3-3061)				
Conduct Reconnaissance by Fire (17-3-0218)				
Follow and Support (17-3-2269)				
Coord/Assist a Passage of Lines Forward/Rearward (17-3-0214)				
Disengage from the enemy (17-3-2380)				
Conduct Deliberate Occupation of a Platoon BP (17-3-2602)				
Conduct Hasty Occupation of a Platoon BP (17-3-2601)				
Conduct a Perimeter Defense (17-3-2632)				
Conduct a Platoon Defense (17-3-2605)				
Conduct a Relief in Place (17-3-1025)				
Displace to a Successive/Alt Platoon BP (17-3-2625)				
BOS: MOBILITY AND SURVIVABILITY				
Conduct Breach Force Operations (17-3-3070)				
Conduct Operational Decontamination (3-3-C016)				
Cross an NBC Contaminated Area (17-3-8143)				
Emplace and Retrieve a Hasty Obstacle (17-3-1026)				
BOS: AIR DEFENSE				
Conduct Passive Air Defense Measures (44-3-C001)				
BOS: COMBAT SERVICE SUPPORT				
Conduct Consolidation and Reorg Activities (12-3-C021)				
Conduct Resupply Operations (17-3-0601)				
BOS: PLATOON BATTLE DRILLS				
Change of Formation Drill (BD-1)				
Contact Drill (BD-2)				
Action Drill (BD-3)				
React to Indirect Fire Drill (BD-4)				
React to Air Attack Drill (BD-5)				
React to a Nuclear Attack Drill (BD-6)				
React to a Chemical/Biological Attack Drill (BD-7)				
TANK GUNNERY				

TRAINING EFFECTS QUESTIONNAIRE: QUARTER 1

ARTEP 17-237-10-MTP TANK PLATOON (1996/07/09)	Starting Prof.	Hours of Field Training	Hours of Sim. Training	Hours of Class. Training	Ending Prof.
BOS: COMMAND AND CONTROL					
Conduct Troop-Leading Procedures (17-3-0065)					
Conduct Assembly Area Activities (17-3-2000)					
Conduct Linkup (17-3-2760)					
BOS: INTELLIGENCE					
Establish Observation Posts (17-3-1039)					
BOS: MANEUVER					
Conduct Bypass Operations (17-3-2420)					
Conduct Convoy Escort (17-3-2320)					
Coord/Conduct a Passage of Lines Forward/Rearward (17-3-1014)					
Conduct Tactical Movement (17-3-1016)					
Conduct a Tactical Road March (17-3-0212)					
Execute Actions on Contact (17-3-0221)					
Destroy an Inferior Force (17-3-2450)					
Assault an Enemy Position (17-3-0220)					
Conduct an Attack by Fire (17-3-0219)					
Conduct Overwatch/Support by Fire (17-3-3061)					
Conduct Reconnaissance by Fire (17-3-0218)					
Follow and Support (17-3-2269)					
Coord/Assist a Passage of Lines Forward/Rearward (17-3-0214)					
Disengage from the enemy (17-3-2380)					
Conduct Deliberate Occupation of a Platoon BP (17-3-2602)					
Conduct Hasty Occupation of a Platoon BP (17-3-2601)					
Conduct a Perimeter Defense (17-3-2632)					
Conduct a Platoon Defense (17-3-2605)					
Conduct a Relief in Place (17-3-1025)					
Displace to a Successive/Alt Platoon BP (17-3-2625)					
BOS: MOBILITY AND SURVIVABILITY					
Conduct Breach Force Operations (17-3-3070)					
Conduct Operational Decontamination (3-3-C016)					
Cross an NBC Contaminated Area (17-3-8143)					
Emplace and Retrieve a Hasty Obstacle (17-3-1026)					
BOS: AIR DEFENSE					
Conduct Passive Air Defense Measures (44-3-C001)					
BOS: COMBAT SERVICE SUPPORT					
Conduct Consolidation and Reorg Activities (12-3-C021)					
Conduct Resupply Operations (17-3-0601)					
BOS: PLATOON BATTLE DRILLS					
Change of Formation Drill (BD-1)					
Contact Drill (BD-2)					
Action Drill (BD-3)					
React to Indirect Fire Drill (BD-4)					
React to Air Attack Drill (BD-5)					
React to a Nuclear Attack Drill (BD-6)					
React to a Chemical/Biological Attack Drill (BD-7)					
TANK GUNNERY					

TRAINING EFFECTS QUESTIONNAIRE: QUARTER 2

ARTEP 17-237-10-MTP TANK PLATOON (1996/07/09)	Hours of Field Training	Hours of Sim. Training	Hours of Class. Training	Ending Prof.
BOS: COMMAND AND CONTROL				
Conduct Troop-Leading Procedures (17-3-0065)				
Conduct Assembly Area Activities (17-3-2000)				
Conduct Linkup (17-3-2760)				
BOS: INTELLIGENCE				
Establish Observation Posts (17-3-1039)				
BOS: MANEUVER				
Conduct Bypass Operations (17-3-2420)				
Conduct Convoy Escort (17-3-2320)				
Coord/Conduct a Passage of Lines Forward/Rearward (17-3-1014)				
Conduct Tactical Movement (17-3-1016)				
Conduct a Tactical Road March (17-3-0212)				
Execute Actions on Contact (17-3-0221)				
Destroy an Inferior Force (17-3-2450)				
Assault an Enemy Position (17-3-0220)				
Conduct an Attack by Fire (17-3-0219)				
Conduct Overwatch/Support by Fire (17-3-3061)				
Conduct Reconnaissance by Fire (17-3-0218)				
Follow and Support (17-3-2269)				
Coord/Assist a Passage of Lines Forward/Rearward (17-3-0214)				
Disengage from the enemy (17-3-2380)				
Conduct Deliberate Occupation of a Platoon BP (17-3-2602)				
Conduct Hasty Occupation of a Platoon BP (17-3-2601)				
Conduct a Perimeter Defense (17-3-2632)				
Conduct a Platoon Defense (17-3-2605)				
Conduct a Relief in Place (17-3-1025)				
Displace to a Successive/Alt Platoon BP (17-3-2625)				
BOS: MOBILITY AND SURVIVABILITY				
Conduct Breach Force Operations (17-3-3070)				
Conduct Operational Decontamination (3-3-C016)				
Cross an NBC Contaminated Area (17-3-8143)				
Emplace and Retrieve a Hasty Obstacle (17-3-1026)				
BOS: AIR DEFENSE				
Conduct Passive Air Defense Measures (44-3-C001)				
BOS: COMBAT SERVICE SUPPORT				
Conduct Consolidation and Reorg Activities (12-3-C021)				
Conduct Resupply Operations (17-3-0601)				
BOS: PLATOON BATTLE DRILLS				
Change of Formation Drill (BD-1)				
Contact Drill (BD-2)				
Action Drill (BD-3)				
React to Indirect Fire Drill (BD-4)				
React to Air Attack Drill (BD-5)				
React to a Nuclear Attack Drill (BD-6)				
React to a Chemical/Biological Attack Drill (BD-7)				
TANK GUNNERY				

TRAINING EFFECTS QUESTIONNAIRE: QUARTER 3

ARTEP 17-237-10-MTP TANK PLATOON (1996/07/09)	Hours of Field Training	Hours of Sim. Training	Hours of Class. Training	Ending Prof.
BOS: COMMAND AND CONTROL				
Conduct Troop-Leading Procedures (17-3-0065)				
Conduct Assembly Area Activities (17-3-2000)				
Conduct Linkup (17-3-2760)				
BOS: INTELLIGENCE				
Establish Observation Posts (17-3-1039)				
BOS: MANEUVER				
Conduct Bypass Operations (17-3-2420)				
Conduct Convoy Escort (17-3-2320)				
Coord/Conduct a Passage of Lines Forward/Rearward (17-3-1014)				
Conduct Tactical Movement (17-3-1016)				
Conduct a Tactical Road March (17-3-0212)				
Execute Actions on Contact (17-3-0221)				
Destroy an Inferior Force (17-3-2450)				
Assault an Enemy Position (17-3-0220)				
Conduct an Attack by Fire (17-3-0219)				
Conduct Overwatch/Support by Fire (17-3-3061)				
Conduct Reconnaissance by Fire (17-3-0218)				
Follow and Support (17-3-2269)				
Coord/Assist a Passage of Lines Forward/Rearward (17-3-0214)				
Disengage from the enemy (17-3-2380)				
Conduct Deliberate Occupation of a Platoon BP (17-3-2602)				
Conduct Hasty Occupation of a Platoon BP (17-3-2601)				
Conduct a Perimeter Defense (17-3-2632)				
Conduct a Platoon Defense (17-3-2605)				
Conduct a Relief in Place (17-3-1025)				
Displace to a Successive/Alt Platoon BP (17-3-2625)				
BOS: MOBILITY AND SURVIVABILITY				
Conduct Breach Force Operations (17-3-3070)				
Conduct Operational Decontamination (3-3-C016)				
Cross an NBC Contaminated Area (17-3-8143)				
Emplace and Retrieve a Hasty Obstacle (17-3-1026)				
BOS: AIR DEFENSE				
Conduct Passive Air Defense Measures (44-3-C001)				
BOS: COMBAT SERVICE SUPPORT				
Conduct Consolidation and Reorg Activities (12-3-C021)				
Conduct Resupply Operations (17-3-0601)				
BOS: PLATOON BATTLE DRILLS				
Change of Formation Drill (BD-1)				
Contact Drill (BD-2)				
Action Drill (BD-3)				
React to Indirect Fire Drill (BD-4)				
React to Air Attack Drill (BD-5)				
React to a Nuclear Attack Drill (BD-6)				
React to a Chemical/Biological Attack Drill (BD-7)				
TANK GUNNERY				

TRAINING EFFECTS QUESTIONNAIRE: QUARTER 4

ARTEP 17-237-10-MTP TANK PLATOON (1996/07/09)	Hours of Field Training	Hours of Sim. Training	Hours of Class. Training	Ending Prof.
BOS: COMMAND AND CONTROL				
Conduct Troop-Leading Procedures (17-3-0065)				
Conduct Assembly Area Activities (17-3-2000)				
Conduct Linkup (17-3-2760)				
BOS: INTELLIGENCE				
Establish Observation Posts (17-3-1039)				
BOS: MANEUVER				
Conduct Bypass Operations (17-3-2420)				
Conduct Convoy Escort (17-3-2320)				
Coord/Conduct a Passage of Lines Forward/Rearward (17-3-1014)				
Conduct Tactical Movement (17-3-1016)				
Conduct a Tactical Road March (17-3-0212)				
Execute Actions on Contact (17-3-0221)				
Destroy an Inferior Force (17-3-2450)				
Assault an Enemy Position (17-3-0220)				
Conduct an Attack by Fire (17-3-0219)				
Conduct Overwatch/Support by Fire (17-3-3061)				
Conduct Reconnaissance by Fire (17-3-0218)				
Follow and Support (17-3-2269)				
Coord/Assist a Passage of Lines Forward/Rearward (17-3-0214)				
Disengage from the enemy (17-3-2380)				
Conduct Deliberate Occupation of a Platoon BP (17-3-2602)				
Conduct Hasty Occupation of a Platoon BP (17-3-2601)				
Conduct a Perimeter Defense (17-3-2632)				
Conduct a Platoon Defense (17-3-2605)				
Conduct a Relief in Place (17-3-1025)				
Displace to a Successive/Alt Platoon BP (17-3-2625)				
BOS: MOBILITY AND SURVIVABILITY				
Conduct Breach Force Operations (17-3-3070)				
Conduct Operational Decontamination (3-3-C016)				
Cross an NBC Contaminated Area (17-3-8143)				
Emplace and Retrieve a Hasty Obstacle (17-3-1026)				
BOS: AIR DEFENSE				
Conduct Passive Air Defense Measures (44-3-C001)				
BOS: COMBAT SERVICE SUPPORT				
Conduct Consolidation and Reorg Activities (12-3-C021)				
Conduct Resupply Operations (17-3-0601)				
BOS: PLATOON BATTLE DRILLS				
Change of Formation Drill (BD-1)				
Contact Drill (BD-2)				
Action Drill (BD-3)				
React to Indirect Fire Drill (BD-4)				
React to Air Attack Drill (BD-5)				
React to a Nuclear Attack Drill (BD-6)				
React to a Chemical/Biological Attack Drill (BD-7)				
TANK GUNNERY				

Appendix B
Building Performance Server Models

This appendix contains information on how to build discrete event simulation models that can be inserted into the Performance Server. The code is specific to the software *Micro Saint*, but the same algorithms can be applied to any simulation software that is COM enabled. Note that this appendix assumes that the user already has basic knowledge on how to build models in *Micro Saint*. The code below includes the specific information that must be added to the models in order for them to be used by TESTIM.

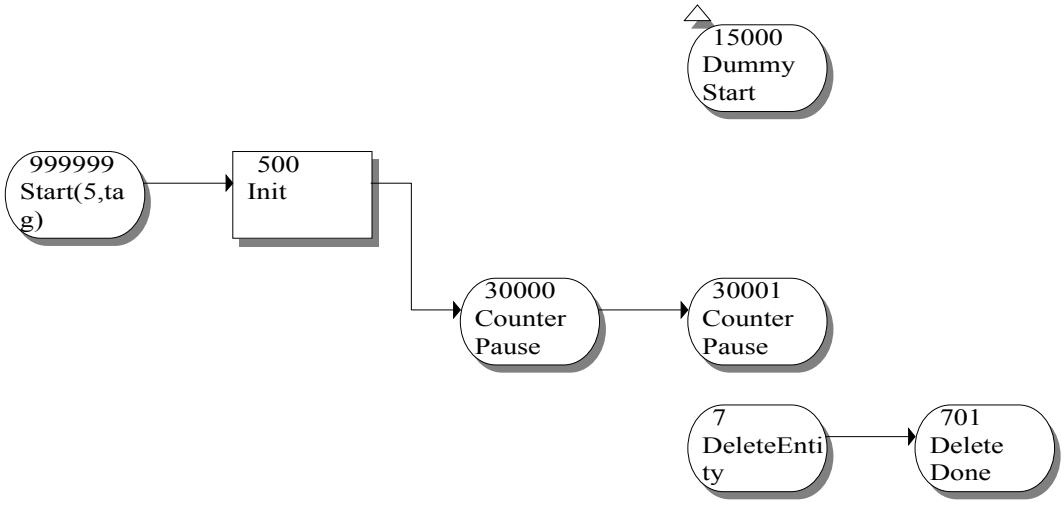

Figure B-1. Base model.

When building a performance server model for the first time, the user must have 1) the basic model (see Figure B-1) to start from that contains tasks that every Performance Server model requires and 2) predefined user functions. From a top-level view, a Performance Server model consists of two parts: the initiating task and the network that contains the Performance Server model (see Figure B-2).

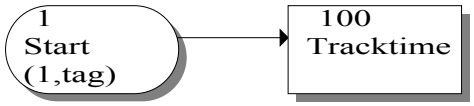

Figure B-2. Performance Server model.

Inside the initiating task, the following code must be entered into the Beginning Effect (see Figure B-3):

```
n += 1;
MAPSTRESSCHK;
```

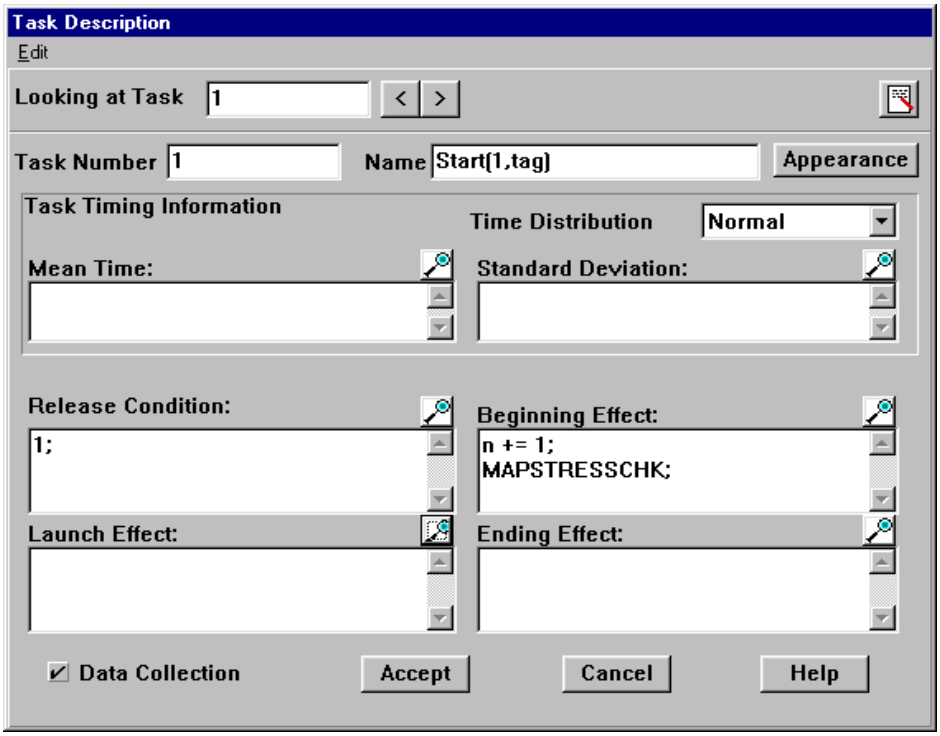

Figure B-3. Beginning Effect of the initiating task.

Inside the performance server model network, each task needs the following code in the Beginning Effect (see Figure B-4):

CurTask := task();

If the performance server model is time based, each task has a time component needs the following code in the Mean Time description box (see Figure B-4):

BaseMean := x; {where x represents the base mean time for this task.}
{A time distribution that the best represents the task, this distribution should use the variable NewMean[CurTask, tag] where the mean time is normally entered.}

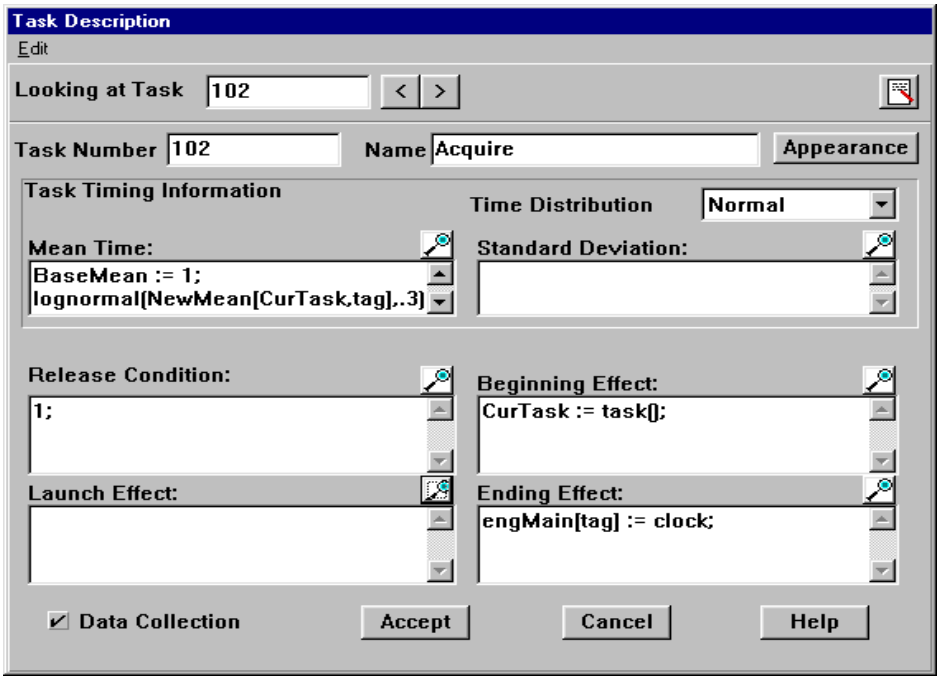

Figure B-4. Server task mean time and beginning effect.

If the performance server model is time based, the user must create an array variable that records the clock time when the model is started. When the model is finished executing, the model must subtract the start time from the clock time to obtain the time it took to perform that Performance Server model. This variable needs to be a single dimensional array (of type real and external) of size 1000, so that the changes in task time can be sent to the Middleware and then TESTIM (see Figure B-5).

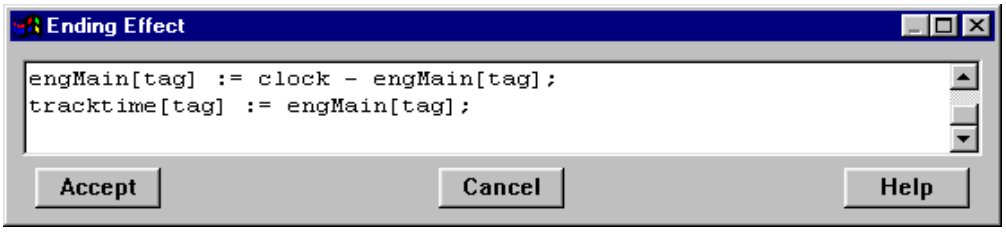

Figure B-5. Performance server model timing code.

When the model has been developed and completed the user must connect the performance server model network to task 30000 (see Figure B-6).

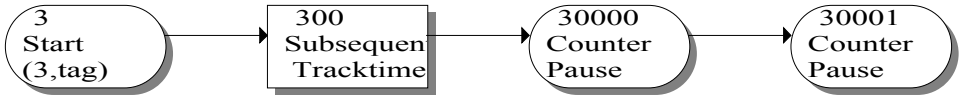

Figure B-6. Full performance server model.